

# HTML

Niveau de vie — Dernière mise à jour le 15  
février 2023

## Table des matières

<b>HTML</b> .....	1
<b>Table des matières</b> .....	1
<b>1 Présentation</b> .....	5
1.1 Où se situe cette spécification ? .....	5
1.2 Est-ce HTML5 ? .....	6
1.3 Contexte .....	6
1.4 Public .....	6
1.5 Portée .....	7
1.6 Historique .....	7
1.7 Notes de conception .....	9
1.8 Syntaxe HTML vs XML .....	11
1.9 Structure de cette spécification .....	12
1.10 Une introduction rapide au HTML .....	15
1.11 Exigences de conformité pour les auteurs .....	23
1.12 Lecture suggérée .....	31
<b>2 Infrastructures communes</b> .....	33
2.1 Terminologie .....	33
2.2 Fonctionnalités contrôlées par des politiques .....	82
2.3 Microsyntaxes courantes .....	82
2.4 URL .....	120
2.5 Récupérer des ressources .....	122
2.6 Interfaces DOM communes .....	132
2.7 Transmission sécurisée de données structurées .....	150
<b>3 Sémantique, structure et API des documents HTML</b> .....	170

3.1 Documents.....	171
3.2 Éléments.....	186
4 Les éléments du HTML .....	242
4.1 L'élément document.....	242
4.2 Métadonnées du document.....	243
4.3 Rubriques.....	295
4.4 Regroupement de contenu.....	330
4.5 Sémantique au niveau du texte .....	369
4.6 Liens.....	428
4.7 Modifications.....	483
4.8 Contenu intégré.....	490
4.9 Données tabulaires .....	719
4.10 Formulaire .....	763
4.11 Éléments interactifs .....	989
4.12 Script.....	1009
4.13 Custom elements.....	1173
4.14 Common idioms without dedicated elements .....	1211
4.15 Disabled elements .....	1219
4.16 Matching HTML elements using selectors and CSS .....	1220
5 Microdata.....	1228
5.1 Introduction .....	1228
5.2 Encodage des microdonnées.....	1236
5.3 Exemples de vocabulaires de microdonnées .....	1247
5.4 Conversion de HTML vers d'autres formats .....	1281
6 Interaction avec l'utilisateur.....	1285
6.1 L' <code>hidden</code> attribut.....	1285
6.2 Visibilité des pages .....	1289
6.3 Sous-arbres inertes.....	1290
6.4 Suivi de l'activation de l'utilisateur .....	1291
6.5 Comportement d'activation des éléments .....	1295
6.6 Mise au point.....	1296
6.7 Attribuer des raccourcis clavier .....	1324
6.8 Édition.....	1328
6.9 Rechercher dans la page .....	1340
6.10 Glisser-déposer.....	1342
6.11 L' <code>popover</code> attribut.....	1372



<b>7 Chargement des pages Web</b>	1387
7.1 Notions complémentaires	1387
7.2 API liées à la navigation et à l'historique des sessions	1422
7.3 Infrastructure pour les séquences de documents	1477
7.4 Historique de navigation et de session	1502
7.5 Cycle de vie des documents	1569
7.6 L' <code>X-Frame-Options</code> en-tête ``	1585
7.7 L' <code>Refresh</code> en-tête ``	1587
7.8 Considérations relatives à l'interface utilisateur du navigateur	1588
<b>8 API d'applications Web</b>	1592
8.1 Script	1592
8.2 Le <code>WindowOrWorkerGlobalScope</code> mixage	1705
8.3 Méthodes utilitaires <code>Base64</code>	1708
8.4 <i>Insertion de balisage dynamique</i>	1709
8.5 Analyse DOM	1715
8.6 Minuteries	1717
8.7 Mise en file d'attente des microtâches	1723
8.8 Invites de l'utilisateur	1726
8.9 État et capacités du système	1730
8.10 Images	1750
8.11 Images animées	1759
<b>9 Communications</b>	1762
9.1 L' <code>MessageEvent</code> interface	1762
9.2 <i>Événements envoyés par le serveur</i>	1765
9.3 <i>Messagerie inter-documents</i>	1781
9.4 <i>Messagerie du canal</i>	1786
9.5 <i>Diffusion vers d'autres contextes de navigation</i>	1797
<b>10 travailleurs du Web</b>	1802
10.1 Présentation	1802
10.2 Infrastructures	1828
10.3 API disponibles pour les travailleurs	1851
<b>11 Worklets</b>	1855
11.1 Présentation	1855
11.2 Exemples	1858
11.3 Infrastructures	1861
<b>12 Stockage Web</b>	1870

12.1 Présentation .....	1870
12.2 L'API .....	1872
12.3 Confidentialité .....	1879
12.4 Sécurité .....	1881
13 La syntaxe <i>HTML</i> .....	1883
13.1 Écrire des documents HTML .....	1883
13.2 Analyser des documents HTML .....	1906
13.3 Sérialisation de fragments HTML .....	2061
13.4 Analyser des fragments HTML .....	2067
13.5 <i>Références de caractères nommés</i> .....	2069
14 La syntaxe <i>XML</i> .....	2132
14.1 Écrire des documents dans la syntaxe XML .....	2132
14.2 Analyser des documents XML .....	2132
14.3 Sérialisation des fragments XML .....	2135
14.4 Analyser des fragments XML .....	2137
15 Rendu .....	2139
15.1 Présentation .....	2139
15.2 La feuille de style de l'agent utilisateur CSS et les conseils de présentation .....	2140
15.3 Éléments non remplacés .....	2142
15.4 Éléments remplacés .....	2181
15.5 Widget .....	2188
15.6 Cadres et jeux de cadres .....	2201
15.7 Médias interactifs .....	2205
15.8 Supports d'impression .....	2209
15.9 Documents XML sans style .....	2209
16 fonctionnalités obsolètes .....	2210
16.1 Caractéristiques obsolètes mais conformes .....	2210
16.2 Éléments non conformes .....	2212
16.3 Exigences pour les implémentations .....	2220
17 Considérations relatives à l'IANA .....	2248
17.1 text/html .....	2248
17.2 multipart/x-mixed-replace .....	2250
17.3 application/xhtml+xml .....	2251
17.4 text/ping .....	2252
17.5 application/microdata+json .....	2254

17.6 text/event-stream .....	2255
17.7 web+ <i>préfixe de schéma</i> .....	2257
Indice .....	2258
Éléments .....	2258
Catégories de contenu d'élément .....	2285
Les attributs .....	2291
Interfaces d'éléments .....	2322
Toutes les interfaces .....	2326
Événements .....	2328
En-têtes HTTP .....	2335
types MIME .....	2335
Les références .....	2337
Remerciements .....	2351
Droits de propriété intellectuelle .....	2356
Table des matières .....	2357
Table des matières complète .....	2358

## 1 Présentation

### 1.1 Où se situe cette spécification ?

Cette spécification définit une grande partie de la plate-forme Web, de manière très détaillée. Sa place dans la pile de spécifications de la plate-forme Web par rapport aux autres spécifications peut être résumée comme suit :

CSS SVG MathML Service Workers IDB Fetch CSP AV1 Opus PNG THIS  
 SPECIFICATION HTTP TLS DOM Unicode Web IDL MIME URL XML JavaScript  
 Encoding

## 1.2 Est-ce HTML5 ?

*Cette section est non normative.*

En bref : Oui.

Plus en détail : le terme "HTML5" est largement utilisé comme mot à la mode pour désigner les technologies Web modernes, dont beaucoup (mais pas toutes) sont développées au WHATWG. Ce document en est un ; d'autres sont disponibles dans [l'aperçu des normes WHATWG](#) .

## 1.3 Contexte

*Cette section est non normative.*

HTML est le langage de balisage de base du World Wide Web. A l'origine, le HTML a été principalement conçu comme un langage permettant de décrire sémantiquement des documents scientifiques. Sa conception générale lui a cependant permis d'être adaptée, au cours des années suivantes, pour décrire un certain nombre d'autres types de documents et même d'applications.

## 1.4 Public

*Cette section est non normative.*

Cette spécification est destinée aux auteurs de documents et de scripts qui utilisent les fonctionnalités définies dans cette spécification, aux implémenteurs d'outils qui fonctionnent sur des pages qui utilisent les fonctionnalités définies dans cette spécification, et aux personnes souhaitant établir l'exactitude des documents ou des implémentations par rapport aux exigences de cette spécification.

Ce document n'est probablement pas adapté aux lecteurs qui n'ont pas déjà au moins une familiarité passagère avec les technologies Web, car par endroits, il sacrifie la clarté à la précision et la brièveté à l'exhaustivité. Des didacticiels et des guides de création plus accessibles peuvent fournir une introduction plus douce au sujet.

En particulier, la familiarité avec les bases de DOM est nécessaire pour une compréhension complète de certaines des parties les plus techniques de cette spécification. Une compréhension de Web IDL, HTTP, XML, Unicode, des encodages de caractères, JavaScript et CSS sera également utile à certains endroits, mais n'est pas essentielle.

## 1.5 Portée

*Cette section est non normative.*

Cette spécification se limite à fournir un langage de balisage de niveau sémantique et des API de script de niveau sémantique associées pour créer des pages accessibles sur le Web allant des documents statiques aux applications dynamiques.

La portée de cette spécification n'inclut pas la fourniture de mécanismes pour la personnalisation de la présentation spécifique au support (bien que des règles de rendu par défaut pour les navigateurs Web soient incluses à la fin de cette spécification, et plusieurs mécanismes pour se connecter à CSS sont fournis dans le cadre du langage).

La portée de cette spécification n'est pas de décrire un système d'exploitation entier. En particulier, les logiciels de configuration matérielle, les outils de manipulation d'images et les applications que les utilisateurs sont censés utiliser quotidiennement avec des postes de travail haut de gamme sont hors de portée. En termes d'applications, cette spécification vise spécifiquement les applications qui devraient être utilisées par les utilisateurs de manière occasionnelle ou régulière mais à partir d'emplacements disparates, avec de faibles exigences en matière de CPU. De telles applications incluent par exemple les systèmes d'achat en ligne, les systèmes de recherche, les jeux (notamment les jeux en ligne multijoueurs), les annuaires téléphoniques publics ou les carnets d'adresses, les logiciels de communication (clients de messagerie, de messagerie instantanée, de discussion), les logiciels d'édition de documents, etc.

## 1.6 Historique

*Cette section est non normative.*

Au cours de ses cinq premières années (1990-1995), HTML a subi un certain nombre de révisions et a connu un certain nombre d'extensions, principalement hébergées d'abord au CERN, puis à l'IETF.

Avec la création du W3C, le développement du HTML a de nouveau changé de lieu. Une première tentative avortée d'extension de HTML en 1995 connue sous le nom de HTML 3.0 a ensuite fait place à une approche plus pragmatique connue sous le nom de HTML 3.2, qui a été achevée en 1997. HTML4 a rapidement suivi plus tard la même année.

L'année suivante, les membres du W3C ont décidé d'arrêter l'évolution du HTML et de commencer à travailler sur un équivalent basé sur XML, appelé XHTML. Cet effort a commencé par une reformulation de HTML4 en XML, connue sous le nom de XHTML 1.0, qui n'a ajouté aucune nouvelle fonctionnalité à l'exception de la nouvelle sérialisation, et qui a été achevée en 2000. Après XHTML 1.0, l'accent du W3C s'est

tourné vers la possibilité pour d'autres groupes de travail de étendre XHTML, sous la bannière de la modularisation XHTML. Parallèlement à cela, le W3C a également travaillé sur un nouveau langage qui n'était pas compatible avec les langages HTML et XHTML antérieurs, l'appelant XHTML2.

À peu près au moment où l'évolution de HTML s'est arrêtée en 1998, des parties de l'API pour HTML développées par les fournisseurs de navigateurs ont été spécifiées et publiées sous le nom de DOM Level 1 (en 1998) et DOM Level 2 Core et DOM Level 2 HTML (à partir de 2000 et culminant en 2003). Ces efforts se sont ensuite essouffés, certaines spécifications DOM niveau 3 étant publiées en 2004, mais le groupe de travail étant fermé avant que tous les projets de niveau 3 ne soient terminés.

En 2003, la publication de XForms, une technologie qui se positionnait comme la prochaine génération de formulaires Web, a suscité un regain d'intérêt pour l'évolution du HTML lui-même, plutôt que pour lui trouver des substituts. Cet intérêt est né de la prise de conscience que le déploiement de XML en tant que technologie Web était limité à des technologies entièrement nouvelles (comme RSS et plus tard Atom), plutôt qu'en remplacement des technologies déployées existantes (comme HTML).

Une preuve de concept pour montrer qu'il était possible d'étendre les formulaires HTML4 pour fournir de nombreuses fonctionnalités introduites par XForms 1.0, sans obliger les navigateurs à implémenter des moteurs de rendu incompatibles avec les pages Web HTML existantes, a été le premier résultat de ce regain d'intérêt. À ce stade précoce, alors que le brouillon était déjà accessible au public et que des contributions étaient déjà sollicitées de toutes les sources, la spécification était uniquement sous le droit d'auteur d'Opera Software.

L'idée que l'évolution de HTML devrait être rouverte a été testée lors d'un atelier du W3C en 2004, où certains des principes qui sous-tendent le travail HTML5 (décrit ci-dessous), ainsi que la première ébauche de proposition susmentionnée couvrant uniquement les fonctionnalités liées aux formulaires, ont été présentés à le W3C conjointement par Mozilla et Opera. La proposition a été rejetée au motif que la proposition était en conflit avec la direction précédemment choisie pour l'évolution du Web; le personnel et les membres du W3C ont voté pour continuer à développer des remplacements basés sur XML à la place.

Peu de temps après, Apple, Mozilla et Opera ont annoncé conjointement leur intention de continuer à travailler sur l'effort sous l'égide d'un nouveau lieu appelé le WHATWG. Une liste de diffusion publique a été créée et le projet a été déplacé sur le site du WHATWG. Le droit d'auteur a ensuite été modifié pour être détenu conjointement par les trois fournisseurs et pour permettre la réutilisation de la spécification.

Le WHATWG était basé sur plusieurs principes fondamentaux, en particulier que les technologies doivent être rétrocompatibles, que les spécifications et les implémentations doivent correspondre même si cela signifie changer la spécification plutôt que les implémentations, et que les spécifications doivent être suffisamment

détaillées pour que les implémentations puissent atteindre interopérabilité complète sans rétro-ingénierie mutuelle.

Cette dernière exigence en particulier exigeait que la portée de la spécification HTML5 inclue ce qui avait été précédemment spécifié dans trois documents distincts : HTML4, XHTML1 et DOM2 HTML. Cela signifiait également inclure beaucoup plus de détails que ce qui était auparavant considéré comme la norme.

En 2006, le W3C a indiqué un intérêt à participer au développement de HTML5 après tout, et en 2007 a formé un groupe de travail chargé de travailler avec le WHATWG sur le développement de la spécification HTML5. Apple, Mozilla et Opera ont autorisé le W3C à publier la spécification sous le copyright du W3C, tout en conservant une version avec la licence la moins restrictive sur le site WHATWG.

Pendant plusieurs années, les deux groupes ont ensuite travaillé ensemble. En 2011, cependant, les groupes sont arrivés à la conclusion qu'ils avaient des objectifs différents : le W3C voulait publier une version "finie" de "HTML5", tandis que le WHATWG voulait continuer à travailler sur un Living Standard pour HTML, en maintenant continuellement la spécification plutôt que de le geler dans un état avec des problèmes connus et d'ajouter de nouvelles fonctionnalités au besoin pour faire évoluer la plate-forme.

En 2019, le WHATWG et le W3C [ont signé un accord](#) pour collaborer sur une version unique de HTML à l'avenir : ce document.

## 1.7 Notes de conception

*Cette section est non normative.*

Il faut admettre que de nombreux aspects du HTML semblent à première vue absurdes et incohérents.

HTML, ses API DOM de support, ainsi que bon nombre de ses technologies de support, ont été développés sur une période de plusieurs décennies par un large éventail de personnes avec des priorités différentes qui, dans de nombreux cas, ne connaissaient pas l'existence les uns des autres.

Les fonctionnalités proviennent donc de nombreuses sources et n'ont pas toujours été conçues de manière particulièrement cohérente. De plus, en raison des caractéristiques uniques du Web, les bogues d'implémentation sont souvent devenus des normes de facto, et maintenant de jure, car le contenu est souvent involontairement écrit d'une manière qui en dépend avant de pouvoir être corrigé.

Malgré tout cela, des efforts ont été faits pour respecter certains objectifs de conception. Ceux-ci sont décrits dans les quelques sous-sections suivantes.

### 1.7.1 Sérialisabilité de l'exécution du script

*Cette section est non normative.*

Pour éviter d'exposer les auteurs Web aux complexités du multithreading, les API HTML et DOM sont conçues de telle sorte qu'aucun script ne puisse jamais détecter l'exécution simultanée d'autres scripts. Même avec [les workers](#), l'intention est que le comportement des implémentations puisse être considéré comme une sérialisation complète de l'exécution de tous les scripts dans tous les globaux.

L'exception à ce principe de conception général est la [SharedArrayBuffer](#) classe JavaScript. En utilisant [SharedArrayBuffer](#) des objets, on peut en effet observer que des scripts dans d'autres [agents](#) s'exécutent simultanément. De plus, en raison du modèle de mémoire JavaScript, il existe des situations non seulement non représentables via l'exécution *de scripts* sérialisés, mais également non représentables via l'exécution *d'instructions* sérialisées parmi ces scripts.

### 1.7.2 Conformité aux autres spécifications

*Cette section est non normative.*

Cette spécification interagit avec et s'appuie sur une grande variété d'autres spécifications. Dans certaines circonstances, malheureusement, des besoins contradictoires ont conduit cette spécification à violer les exigences de ces autres spécifications. Chaque fois que cela s'est produit, les transgressions ont chacune été notées comme une " **violation délibérée** ", et la raison de la violation a été notée.

### 1.7.3 Extensibilité

*Cette section est non normative.*

HTML dispose d'un large éventail de mécanismes d'extensibilité qui peuvent être utilisés pour ajouter de la sémantique de manière sûre :

- Les auteurs peuvent utiliser l' [class](#) attribut pour étendre des éléments, en créant efficacement leurs propres éléments, tout en utilisant l'élément HTML "réel" existant le plus applicable, de sorte que les navigateurs et autres outils qui ne connaissent pas l'extension puissent toujours la prendre en charge assez bien. C'est la tactique utilisée par les microformats, par exemple.
- Les auteurs peuvent inclure des données pour les scripts côté client en ligne ou les scripts côté serveur à l'échelle du site à traiter à l'aide des [data-\\*](#) attributs. Ceux-ci sont garantis de ne jamais être touchés par les



navigateurs et permettent aux scripts d'inclure des données sur les éléments HTML que les scripts peuvent ensuite rechercher et traiter.

- Les auteurs peuvent utiliser le `<meta name="" content="">` mécanisme pour inclure des métadonnées à l'échelle de la page.
- Les auteurs peuvent utiliser le `rel=""` mécanisme pour annoter des liens avec des significations spécifiques en enregistrant [des extensions dans l'ensemble prédéfini de types de liens](#) . Ceci est également utilisé par les microformats.
- Les auteurs peuvent intégrer des données brutes à l'aide du `<script type="">` mécanisme avec un type personnalisé, pour une manipulation ultérieure par des scripts en ligne ou côté serveur.
- Les auteurs peuvent étendre les API à l'aide du mécanisme de prototypage JavaScript. Ceci est largement utilisé par les bibliothèques de scripts, par exemple.
- Les auteurs peuvent utiliser la fonction de microdonnées (les attributs `itemscope=""` et `itemprop=""` ) pour intégrer des paires nom-valeur imbriquées de données à partager avec d'autres applications et sites.

## 1.8 Syntaxe HTML vs XML

*Cette section est non normative.*

Cette spécification définit un langage abstrait pour décrire les documents et les applications, et certaines API pour interagir avec les représentations en mémoire des ressources qui utilisent ce langage.

La représentation en mémoire est connue sous le nom de "DOM HTML", ou "le DOM" en abrégé.

Il existe diverses syntaxes concrètes qui peuvent être utilisées pour transmettre des ressources qui utilisent ce langage abstrait, dont deux sont définies dans la présente spécification.

La première de ces syntaxes concrètes est la syntaxe HTML. C'est le format suggéré pour la plupart des auteurs. Il est compatible avec la plupart des navigateurs Web hérités. Si un document est transmis avec le `text/html` [type MIME](#) , il sera traité comme un document HTML par les navigateurs Web. Cette spécification définit la dernière syntaxe HTML, appelée simplement "HTML".

La deuxième syntaxe concrète est XML. Lorsqu'un document est transmis avec un [type XML MIME](#) , tel que `application/xhtml+xml`, il est alors traité comme un document XML par les navigateurs Web, pour être analysé par un processeur XML. Il est rappelé aux auteurs que le traitement pour XML et HTML diffère ; en particulier,

même des erreurs de syntaxe mineures empêcheront un document étiqueté comme XML d'être entièrement restitué, alors qu'elles seraient ignorées dans la syntaxe HTML.

*La syntaxe XML pour HTML était auparavant appelée "XHTML", mais cette spécification n'utilise pas ce terme (entre autres raisons, car aucun terme de ce type n'est utilisé pour les syntaxes HTML de MathML et SVG).*

Le DOM, la syntaxe HTML et la syntaxe XML ne peuvent pas tous représenter le même contenu. Par exemple, les espaces de noms ne peuvent pas être représentés à l'aide de la syntaxe HTML, mais ils sont pris en charge dans le DOM et dans la syntaxe XML. De même, les documents qui utilisent la `noscript` fonctionnalité peuvent être représentés à l'aide de la syntaxe HTML, mais ne peuvent pas être représentés avec le DOM ou dans la syntaxe XML. Les commentaires contenant la chaîne " `-->`" ne peuvent être représentés que dans le DOM, pas dans les syntaxes HTML et XML.

## 1.9 Structure de cette spécification

*Cette section est non normative.*

Cette spécification est divisée en grandes sections suivantes :

### Introduction

Documents non normatifs fournissant un contexte pour la norme HTML.

### Infrastructure commune

Les classes de conformité, les algorithmes, les définitions et les fondements communs du reste de la spécification.

### Sémantique, structure et API des documents HTML

Les documents sont construits à partir d'éléments. Ces éléments forment un arbre en utilisant le DOM. Cette section définit les fonctionnalités de ce DOM, ainsi que l'introduction des fonctionnalités communes à tous les éléments, et les concepts utilisés dans la définition des éléments.

### Les éléments du HTML

Chaque élément a une signification prédéfinie, qui est expliquée dans cette section. Des règles pour les auteurs sur la façon d'utiliser l'élément, ainsi que les exigences de l'agent utilisateur sur la façon de gérer chaque élément, sont également données. Cela inclut de grandes fonctionnalités de signature HTML telles que la lecture vidéo et les sous-titres, les contrôles de formulaire et la soumission de formulaire, ainsi qu'une API graphique 2D connue sous le nom de canevas HTML.

### Microdonnées

Cette spécification introduit un mécanisme pour ajouter des annotations lisibles par machine aux documents, afin que les outils puissent extraire des arborescences de paires nom-valeur du document. Cette section décrit ce mécanisme et certains algorithmes qui peuvent être utilisés pour convertir des documents HTML dans d'autres formats. Cette section définit également quelques exemples de vocabulaires de microdonnées pour les informations de contact, les événements du calendrier et les travaux de licence.

### **Interaction de l'utilisateur**

Les documents HTML peuvent fournir un certain nombre de mécanismes permettant aux utilisateurs d'interagir avec le contenu et de le modifier, qui sont décrits dans cette section, tels que le fonctionnement du focus et le glisser-déposer.

### **Chargement des pages Web**

Les documents HTML n'existent pas dans le vide - cette section définit de nombreuses fonctionnalités qui affectent les environnements traitant de plusieurs pages, tels que les navigateurs Web.

### **API d'applications Web**

Cette section présente les fonctionnalités de base pour la création de scripts d'applications en HTML.

### **Travailleurs Web**

Cette section définit une API pour les threads d'arrière-plan en JavaScript.

### **Worklets**

Cette section définit l'infrastructure des API qui doivent exécuter JavaScript séparément de l'environnement d'exécution JavaScript principal.

### **Les API de communication**

Cette section décrit certains mécanismes que les applications écrites en HTML peuvent utiliser pour communiquer avec d'autres applications de différents domaines s'exécutant sur le même client. Il introduit également un mécanisme de flux d'événements push serveur appelé Server Sent Events ou [EventSource](#), et un protocole de socket bidirectionnel en duplex intégral pour les scripts appelé Web Sockets.

### **espace archivage sur le Web**

Cette section définit un mécanisme de stockage côté client basé sur des paires nom-valeur.

### **La syntaxe HTML**

### **La syntaxe XML**

Toutes ces fonctionnalités seraient inutiles si elles ne pouvaient pas être représentées sous une forme sérialisée et envoyées à d'autres personnes. Ces sections définissent donc les syntaxes de HTML et XML, ainsi que des règles sur la façon d'analyser le contenu à l'aide de ces syntaxes.

### **Le rendu**

Cette section définit les règles de rendu par défaut pour les navigateurs Web.

Il existe également des annexes, répertoriant [les fonctionnalités obsolètes](#) et [les considérations IANA](#) , ainsi que plusieurs index.

### 1.9.1 Comment lire cette spécification

Cette spécification doit être lue comme toutes les autres spécifications. Tout d'abord, il doit être lu d'un bout à l'autre, plusieurs fois. Ensuite, il doit être lu à l'envers au moins une fois. Ensuite, il doit être lu en choisissant des sections aléatoires dans la liste de contenu et en suivant toutes les références croisées.

Comme décrit dans la section des exigences de conformité ci-dessous, cette spécification décrit les critères de conformité pour une variété de classes de conformité. En particulier, il existe des exigences de conformité qui s'appliquent aux *producteurs* , par exemple les auteurs et les documents qu'ils créent, et il existe des exigences de conformité qui s'appliquent aux *consommateurs* , par exemple les navigateurs Web. Ils peuvent être distingués par ce qu'ils exigent : une exigence imposée à un producteur indique ce qui est autorisé, tandis qu'une exigence imposée à un consommateur indique comment le logiciel doit agir.

Par exemple, "la `foo` valeur de l'attribut doit être un [entier valide](#) " est une exigence imposée aux producteurs, car elle définit les valeurs autorisées ; en revanche, l'exigence "la `foo` valeur de l'attribut doit être analysée à l'aide des [règles d'analyse des entiers](#) " est une exigence pour les consommateurs, car elle décrit comment traiter le contenu.

#### Les exigences imposées aux producteurs n'ont aucune incidence sur les consommateurs.

Poursuivant l'exemple ci-dessus, une exigence indiquant que la valeur d'un attribut particulier est contrainte d'être un [entier valide](#) *n'implique* absolument rien sur les exigences imposées aux consommateurs. Il se peut que les consommateurs soient en fait tenus de traiter l'attribut comme une chaîne opaque, totalement indépendante du fait que la valeur soit conforme ou non aux exigences. Il se peut (comme dans l'exemple précédent) que les consommateurs soient tenus d'analyser la valeur à l'aide de règles spécifiques qui définissent la manière dont les valeurs non valides (non numériques dans ce cas) doivent être traitées.

### 1.9.2 Conventions typographiques

Il s'agit d'une définition, d'une exigence ou d'une explication.

*Ceci est une remarque.*

Ceci est un exemple.

C'est un problème ouvert.

**Ceci est un avertissement.**

```
[Exposed=Window]
```

```
interface Example {
```

```
// this is an IDL definition
```

```
};
```

**`variable = object.method([optionalArgument])`**

**Ceci est une note aux auteurs décrivant l'utilisation d'une interface.**

```
/* this is a CSS fragment */
```

L'instance de définition d'un terme est balisée comme **ceci** . Les utilisations de ce terme sont balisées comme [ceci](#) ou comme [ceci](#) .

L'instance de définition d'un élément, d'un attribut ou d'une API est balisée comme **this**. Les références à cet élément, attribut ou API sont balisées comme [this](#).

D'autres fragments de code sont balisés `like this`.

Les variables sont marquées comme *ceci* .

Dans un algorithme, les étapes des [sections synchrones](#) sont marquées par ⌚.

Dans certains cas, les exigences sont données sous forme de listes avec les conditions et les exigences correspondantes. Dans de tels cas, les exigences qui s'appliquent à une condition sont toujours le premier ensemble d'exigences qui suit la condition, même dans le cas où il existe plusieurs ensembles de conditions pour ces exigences. Ces cas se présentent comme suit :

**C'est une condition**

**C'est une autre condition**

C'est l'exigence qui s'applique aux conditions ci-dessus.

**C'est une troisième condition**

C'est l'exigence qui s'applique à la troisième condition.

## 1.10 Une introduction rapide au HTML

*Cette section est non normative.*

Un document HTML de base ressemble à ceci :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Sample page</title>
  </head>
  <body>
    <h1>Sample page</h1>
    <p>This is a <a href="demo.html">simple</a> sample.</p>
    <!-- this is a comment -->
  </body>
</html>
```

Les documents HTML consistent en une arborescence d'éléments et de texte. Chaque élément est désigné dans la source par une [balise de début](#), telle que "`<body>`", et une [balise de fin](#), telle que "`</body>`". (Certaines balises de début et certaines balises de fin peuvent dans certains cas être [omises](#) et sont impliquées par d'autres balises.)

Les balises doivent être imbriquées de manière à ce que les éléments soient tous complètement les uns dans les autres, sans se chevaucher :

```
<p>This is <em>very <strong>wrong</strong>!</strong></p>
<p>This <em>is <strong>correct</strong>.</em></p>
```

Cette spécification définit un ensemble d'éléments pouvant être utilisés en HTML, ainsi que des règles sur la manière dont les éléments peuvent être imbriqués.

Les éléments peuvent avoir des attributs, qui contrôlent le fonctionnement des éléments. Dans l'exemple ci-dessous, il y a un [lien hypertexte](#), formé à l'aide de l'[a](#) élément et de son [href](#) attribut :

```
<a href="demo.html">simple</a>
```

[Les attributs](#) sont placés à l'intérieur de la balise de début et se composent d'un [nom](#) et d'une [valeur](#), séparés par un caractère " ". La valeur de l'attribut peut rester [sans guillemets](#) si elle ne contient pas [d'espaces blancs ASCII](#) ou l'un de " ' ` = <ou >. Sinon, il doit être cité à l'aide de guillemets simples ou doubles. La valeur, ainsi que le caractère " ", peuvent être complètement omis si la valeur est la chaîne vide.

```
<!-- empty attributes -->
<input name=address disabled>
<input name=address disabled="">
```

```

<!-- attributes with a value -->
<input name=address maxlength=200>
<input name=address maxlength='200'>
<input name=address maxlength="200">

```

Les agents utilisateurs HTML (par exemple, les navigateurs Web) *analysent* ensuite ce balisage, le transformant en un arbre DOM (Document Object Model). Un arbre DOM est une représentation en mémoire d'un document.

Les arbres DOM contiennent plusieurs sortes de nœuds, notamment un DocumentType nœud, Element des nœuds, Text des nœuds, Comment des nœuds et, dans certains cas, ProcessingInstruction des nœuds.

L' [extrait de balisage en haut de cette section](#) serait transformé en l'arborescence DOM suivante :

- DOCTYPE :html
- html lang=" en "
  - head
    - #text: ↵ \_ \_
    - title
      - #text: *Exemple de page*
    - #text: ↵ \_
  - #text: ↵ \_
  - body
    - #text: ↵ \_ \_
    - h1
      - #text: *Exemple de page*
    - #text: ↵ \_ \_
    - p
      - #text: *C'est un*
      - a href=" demo.html "
        - #text: *simples*
      - #text: *échantillon.*
    - #text: ↵ \_ \_
    - #comment: *c'est un commentaire*
    - #text: ↵ \_ ↵

L' [élément document](#) de cette arborescence est l' html élément, qui est l'élément qui se trouve toujours à cette position dans les documents HTML. Il contient deux éléments, head et body, ainsi qu'un Text nœud entre eux.

Il y a beaucoup plus Text de nœuds dans l'arborescence DOM que ce à quoi on pourrait s'attendre initialement, car la source contient un certain nombre d'espaces (représentés ici par " \_ ") et de sauts de ligne (" ↵ ") qui finissent tous comme des nœuds dans le DOM Text. Cependant, pour des raisons historiques, tous les espaces et les sauts de ligne du balisage d'origine n'apparaissent pas dans le

DOM. En particulier, tous les espaces blancs avant `head` la balise de début finissent par être supprimés silencieusement, et tous les espaces blancs après la `body` balise de fin finissent par être placés à la fin du `body`.

L' `head` élément contient un `title` élément, qui contient lui-même un `Text` nœud avec le texte "Sample page". De même, l' `body` élément contient un `h1` élément, un `p` élément et un commentaire.

---

Cet arbre DOM peut être manipulé à partir de scripts dans la page. Les scripts (généralement en JavaScript) sont de petits programmes qui peuvent être intégrés à l'aide de l' `script` élément ou à l'aide [des attributs de contenu du gestionnaire d'événements](#) . Par exemple, voici un formulaire avec un script qui définit la valeur de l' `output` élément du formulaire pour dire "Hello World":

```
<form name="main">
  Result: <output name="result"></output>
  <script>
    document.forms.main.elements.result.value = 'Hello World';
  </script>
</form>
```

Chaque élément de l'arborescence DOM est représenté par un objet, et ces objets ont des API afin qu'ils puissent être manipulés. Par exemple, un lien (par exemple l' `a` élément dans l'arborescence ci-dessus) peut voir son `href` attribut " " modifié de plusieurs manières :

```
var a = document.links[0]; // obtain the first link in the document
a.href = 'sample.html'; // change the destination URL of the link
a.protocol = 'https'; // change just the scheme part of the URL
a.setAttribute('href', 'https://example.com/'); // change the
content attribute directly
```

Étant donné que les arbres DOM sont utilisés pour représenter les documents HTML lorsqu'ils sont traités et présentés par les implémentations (en particulier les implémentations interactives comme les navigateurs Web), cette spécification est principalement formulée en termes d'arbres DOM, au lieu du balisage décrit ci-dessus.

---



Les documents HTML représentent une description indépendante du média du contenu interactif. Les documents HTML peuvent être affichés sur un écran, via un synthétiseur vocal ou sur un afficheur braille. Pour influencer exactement la façon dont un tel rendu a lieu, les auteurs peuvent utiliser un langage de style tel que CSS.

Dans l'exemple suivant, la page a été rendue jaune sur bleu à l'aide de CSS.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Sample styled page</title>
    <style>
      body { background: navy; color: yellow; }
    </style>
  </head>
  <body>
    <h1>Sample styled page</h1>
    <p>This page is just a demo.</p>
  </body>
</html>
```

Pour plus de détails sur l'utilisation du HTML, les auteurs sont encouragés à consulter des didacticiels et des guides. Certains des exemples inclus dans cette spécification peuvent également être utiles, mais l'auteur novice est averti que cette spécification, par nécessité, définit le langage avec un niveau de détail qui peut être difficile à comprendre au début.

### 1.10.1 Écrire des applications sécurisées avec HTML

*Cette section est non normative.*

Lorsque HTML est utilisé pour créer des sites interactifs, il faut veiller à éviter d'introduire des vulnérabilités par lesquelles des attaquants peuvent compromettre l'intégrité du site lui-même ou des utilisateurs du site.

Une étude approfondie de cette question est au-delà de la portée de ce document, et les auteurs sont fortement encouragés à étudier la question plus en détail. Cependant, cette section tente de fournir une introduction rapide à certains pièges courants dans le développement d'applications HTML.

Le modèle de sécurité du Web est basé sur le concept des "origines", et par conséquent, de nombreuses attaques potentielles sur le Web impliquent des actions inter-origines. [\[ORIGINE\]](#)

## Ne valide pas l'entrée de l'utilisateur

### Script intersite (XSS)

#### Injection SQL

Lors de l'acceptation d'entrées non fiables, par exemple du contenu généré par l'utilisateur tel que des commentaires textuels, des valeurs dans les paramètres d'URL, des messages de sites tiers, etc., il est impératif que les données soient validées avant utilisation et correctement échappées lorsqu'elles sont affichées. Ne pas le faire peut permettre à un utilisateur hostile d'effectuer une variété d'attaques, allant des attaques potentiellement bénignes, comme fournir de fausses informations sur l'utilisateur comme un âge négatif, aux plus sérieuses, comme exécuter des scripts chaque fois qu'un utilisateur regarde une page qui inclut les informations, propageant potentiellement l'attaque dans le processus, au catastrophique, comme la suppression de toutes les données du serveur.

Lors de l'écriture de filtres pour valider l'entrée de l'utilisateur, il est impératif que les filtres soient toujours basés sur des listes sûres, en autorisant les constructions sûres connues et en interdisant toutes les autres entrées. Les filtres basés sur des listes de blocage qui interdisent les entrées connues mauvaises et autorisent tout le reste ne sont pas sécurisés, car tout ce qui est mauvais n'est pas encore connu (par exemple, parce qu'il pourrait être inventé à l'avenir).

Par exemple, supposons qu'une page regarde la chaîne de requête de son URL pour déterminer ce qu'elle doit afficher, et que le site redirige ensuite l'utilisateur vers cette page pour afficher un message, comme dans :

```
<ul>
  <li><a href="message.cgi?say=Hello">Say Hello</a>
  <li><a href="message.cgi?say=Welcome">Say Welcome</a>
  <li><a href="message.cgi?say=Kittens">Say Kittens</a>
</ul>
```

Si le message était simplement affiché à l'utilisateur sans s'échapper, un attaquant hostile pourrait alors créer une URL contenant un élément de script :

```
https://example.com/message.cgi?say=%3Cscript%3Ealert%28%27Oh
%20no%21%27%29%3C/script%3E
```

Si l'attaquant convainquait ensuite un utilisateur victime de visiter cette page, un script choisi par l'attaquant s'exécuterait sur la page. Un tel script pourrait faire un certain nombre d'actions hostiles, limitées uniquement par ce que le site propose : si le site est une boutique de commerce électronique, par exemple, un tel script pourrait amener l'utilisateur à effectuer sans le savoir de nombreux achats non désirés.

C'est ce qu'on appelle une attaque de script intersite.

Il existe de nombreuses constructions qui peuvent être utilisées pour tenter d'inciter un site à exécuter du code. En voici quelques-uns que les auteurs

sont encouragés à prendre en compte lors de l'écriture de filtres de listes sûres :

- Lorsque vous autorisez des éléments apparemment inoffensifs tels que `img`, il est également important de mettre sur liste sûre tous les attributs fournis. Si l'on autorisait tous les attributs, un attaquant pourrait, par exemple, utiliser l' `onload` attribut pour exécuter un script arbitraire.
- Lorsque vous autorisez la fourniture d'URL (par exemple pour des liens), le schéma de chaque URL doit également être explicitement mis sur liste sûre, car de nombreux schémas peuvent être utilisés de manière abusive. L'exemple le plus frappant est "`javascript:`", mais les agents utilisateurs peuvent en implémenter (et en ont historiquement implémenté) d'autres.
- `base` Autoriser l'insertion d' un élément signifie que tous `script` les éléments de la page avec des liens relatifs peuvent être piratés, et de même que toute soumission de formulaire peut être redirigée vers un site hostile.

### Contrefaçon de requête intersite (CSRF)

Si un site permet à un utilisateur de soumettre des formulaires avec des effets secondaires spécifiques à l'utilisateur, par exemple en publiant des messages sur un forum sous le nom de l'utilisateur, en effectuant des achats ou en demandant un passeport, il est important de vérifier que la demande a été faite par l'utilisateur intentionnellement, plutôt que par un autre site incitant l'utilisateur à faire la demande sans le savoir.

Ce problème existe car les formulaires HTML peuvent être soumis à d'autres origines.

Les sites peuvent empêcher de telles attaques en remplissant les formulaires avec des jetons masqués spécifiques à l'utilisateur ou en vérifiant `Origin` les en-têtes `` sur toutes les requêtes.

### Clicjacking

Une page qui fournit aux utilisateurs une interface pour effectuer des actions que l'utilisateur ne souhaite peut-être pas effectuer doit être conçue de manière à éviter la possibilité que les utilisateurs puissent être amenés à activer l'interface.

Un utilisateur pourrait être trompé si un site hostile place le site victime dans un petit espace `iframe` et convainc ensuite l'utilisateur de cliquer, par exemple en le faisant jouer à un jeu de réaction. Une fois que l'utilisateur joue au jeu, le site hostile peut rapidement positionner l'iframe sous le curseur de la souris juste au moment où l'utilisateur est sur le point de cliquer, incitant ainsi l'utilisateur à cliquer sur l'interface du site victime.

Pour éviter cela, les sites qui ne prévoient pas d'être utilisés dans des cadres sont encouragés à n'activer leur interface que s'ils détectent qu'ils ne sont pas dans un cadre (par exemple en comparant l'objet à la valeur de l'attribut `window.top`).

### 1.10.2 Pièges courants à éviter lors de l'utilisation des API de script

*Cette section est non normative.*

Les scripts en HTML ont une sémantique "d'exécution jusqu'à la fin", ce qui signifie que le navigateur exécutera généralement le script sans interruption avant de faire quoi que ce soit d'autre, comme déclencher d'autres événements ou continuer à analyser le document.

D'autre part, l'analyse des fichiers HTML se fait de manière incrémentielle, ce qui signifie que l'analyseur peut s'arrêter à tout moment pour laisser les scripts s'exécuter. C'est généralement une bonne chose, mais cela signifie que les auteurs doivent faire attention à ne pas accrocher les gestionnaires d'événements après que les événements auraient pu se déclencher.

Il existe deux techniques pour effectuer cette opération de manière fiable : utilisez [les attributs de contenu du gestionnaire d'événements](#) ou créez l'élément et ajoutez les gestionnaires d'événements dans le même script. Ce dernier est sûr car, comme mentionné précédemment, les scripts sont exécutés jusqu'à la fin avant que d'autres événements ne puissent se déclencher.

Une façon dont cela pourrait se manifester est avec `img` les éléments et l' `load` événement. L'événement peut se déclencher dès que l'élément a été analysé, en particulier si l'image a déjà été mise en cache (ce qui est courant).

Ici, l'auteur utilise le `onload` gestionnaire sur un `img` élément pour intercepter l' `load` événement :

```

```

Si l'élément est ajouté par script, tant que les gestionnaires d'événements sont ajoutés dans le même script, l'événement ne sera toujours pas manqué :

```
<script>
var img = new Image();
img.src = 'games.png';
img.alt = 'Games';
img.onload = gamesLogoHasLoaded;
// img.addEventListener('load', gamesLogoHasLoaded, false); //
would work also
```

```
</script>
```

Cependant, si l'auteur a d'abord créé l' `img` élément, puis ajouté les écouteurs d'événement dans un script séparé, il est possible que l' `load` événement soit déclenché entre-temps, ce qui le ferait manquer :

```
<!-- Do not use this style, it has a race condition! -->

<!-- the 'load' event might fire here while the parser is taking a
      break, in which case you will not see it! -->
<script>
  var img = document.getElementById('games');
  img.onload = gamesLogoHasLoaded; // might never fire!
</script>
```

### 1.10.3 Comment détecter les erreurs lors de l'écriture HTML : validateurs et vérificateurs de conformité

*Cette section est non normative.*

Les auteurs sont encouragés à utiliser des vérificateurs de conformité (également appelés *validateurs*) pour détecter les erreurs courantes. Le WHATWG tient à jour une liste de ces outils sur : <https://whatwg.org/validator/>

## 1.11 Exigences de conformité pour les auteurs

*Cette section est non normative.*

Contrairement aux versions précédentes de la spécification HTML, cette spécification définit en détail le traitement requis pour les documents non valides ainsi que pour les documents valides.

Cependant, même si le traitement du contenu invalide est dans la plupart des cas bien défini, les exigences de conformité des documents sont toujours importantes : en pratique, l'interopérabilité (situation dans laquelle toutes les implémentations traitent un contenu particulier de manière fiable et identique ou équivalente) n'est pas le seul objectif des exigences de conformité des documents. Cette section détaille certaines des raisons les plus courantes pour toujours faire la distinction entre un document conforme et un document contenant des erreurs.

### 1.11.1 Balisage de présentation

*Cette section est non normative.*

La majorité des fonctionnalités de présentation des versions précédentes de HTML ne sont plus autorisées. Le balisage de présentation en général présente un certain nombre de problèmes :

#### **L'utilisation d'éléments de présentation conduit à une moins bonne accessibilité**

Bien qu'il soit possible d'utiliser le balisage de présentation d'une manière qui offre aux utilisateurs de technologies d'assistance (AT) une expérience acceptable (par exemple en utilisant ARIA), cela est beaucoup plus difficile que de le faire en utilisant un balisage sémantiquement approprié. De plus, même l'utilisation de ces techniques n'aide pas à rendre les pages accessibles aux utilisateurs non graphiques non AT, tels que les utilisateurs de navigateurs en mode texte.

L'utilisation d'un balisage indépendant du support, d'autre part, offre un moyen simple de créer des documents de manière à ce qu'ils fonctionnent pour un plus grand nombre d'utilisateurs (par exemple, les utilisateurs de navigateurs de texte).

#### **Coût de maintenance plus élevé**

Il est beaucoup plus facile de maintenir un site écrit de manière à ce que le balisage soit indépendant du style. Par exemple, changer la couleur d'un site qui utilise `<font color="">` partout nécessite des changements sur l'ensemble du site, alors qu'un changement similaire sur un site basé sur CSS peut être effectué en modifiant un seul fichier.

#### **Formats de documents plus grands**

Le balisage de présentation a tendance à être beaucoup plus redondant et se traduit donc par des tailles de document plus grandes.

Pour ces raisons, le balisage de présentation a été supprimé du HTML dans cette version. Ce changement ne devrait pas surprendre ; HTML4 a déprécié le balisage de présentation il y a de nombreuses années et a fourni un mode (HTML4 Transitional) pour aider les auteurs à s'éloigner du balisage de présentation ; plus tard, XHTML 1.1 est allé plus loin et a complètement rendu ces fonctionnalités obsolètes.

Les seules fonctionnalités de balisage de présentation restantes en HTML sont l'`style`attribut et l'`style`élément. L'utilisation de l'`style`attribut est quelque peu déconseillée dans les environnements de production, mais il peut être utile pour le prototypage rapide (où ses règles peuvent être directement déplacées dans une feuille de style séparée plus tard) et pour fournir des styles spécifiques dans des cas inhabituels où une feuille de style séparée serait gênante . De même, l'`style`élément peut être utile dans la syndication ou pour des styles spécifiques à une page, mais

en général, une feuille de style externe est susceptible d'être plus pratique lorsque les styles s'appliquent à plusieurs pages.

Il convient également de noter que certains éléments qui étaient auparavant de présentation ont été redéfinis dans cette spécification pour être indépendants du support : [b](#), [i](#), [hr](#), [s](#), [small](#), et [u](#).

### 1.11.2 Erreurs de syntaxe

*Cette section est non normative.*

La syntaxe du HTML est contrainte pour éviter une grande variété de problèmes.

#### Comportement de gestion des erreurs non intuitif

Certaines constructions de syntaxe non valides, lorsqu'elles sont analysées, entraînent des arbres DOM très peu intuitifs.

Par exemple, le fragment de balisage suivant génère un DOM avec un [hr](#)élément qui est un frère *antérieur* de l' [table](#)élément correspondant :

```
<table><hr>...
```

#### Erreurs avec récupération d'erreur facultative

Pour permettre aux agents utilisateurs d'être utilisés dans des environnements contrôlés sans avoir à implémenter les règles de gestion des erreurs les plus bizarres et alambiquées, les agents utilisateurs sont autorisés à échouer chaque fois qu'ils rencontrent une [erreur d'analyse](#) .

#### Erreurs où le comportement de gestion des erreurs n'est pas compatible avec les agents utilisateurs en streaming

Certains comportements de gestion des erreurs, tels que le comportement de l' `<table><hr>...` exemple mentionné ci-dessus, sont incompatibles avec les agents utilisateurs en continu (agents utilisateurs qui traitent les fichiers HTML en une seule passe, sans état de stockage). Pour éviter les problèmes d'interopérabilité avec de tels agents utilisateurs, toute syntaxe entraînant un tel comportement est considérée comme invalide.

#### Erreurs pouvant entraîner une contrainte d'infoset

Lorsqu'un agent utilisateur basé sur XML est connecté à un analyseur HTML, il est possible que certains invariants appliqués par XML, tels que les noms d'éléments ou d'attributs ne contiennent jamais plusieurs points, soient violés par un fichier HTML. La gestion de cela peut nécessiter que l'analyseur contraigne le DOM HTML dans un ensemble d'informations compatible XML. La plupart des constructions de syntaxe qui nécessitent un tel traitement sont considérées comme non valides. (Les commentaires contenant deux

traits d'union consécutifs ou se terminant par un trait d'union sont des exceptions autorisées dans la syntaxe HTML.)

## Erreurs entraînant des performances disproportionnellement médiocres

Certaines constructions de syntaxe peuvent entraîner des performances disproportionnées. Pour décourager l'utilisation de telles constructions, elles sont généralement rendues non conformes.

Par exemple, le balisage suivant entraîne de mauvaises performances, car tous les i éléments non fermés doivent être reconstruits dans chaque paragraphe, ce qui entraîne progressivement plus d'éléments dans chaque paragraphe :

```
<p><i>She dreamt.  
<p><i>She dreamt that she ate breakfast.  
<p><i>Then lunch.  
<p><i>And finally dinner.
```

Le DOM résultant pour ce fragment serait :



## Erreurs impliquant des constructions de syntaxe fragiles

Il existe des constructions syntaxiques qui, pour des raisons historiques, sont relativement fragiles. Pour aider à réduire le nombre d'utilisateurs qui rencontrent accidentellement de tels problèmes, ils sont rendus non conformes.

Par exemple, l'analyse de certaines références de caractères nommés dans les attributs se produit même si le point-virgule de fermeture est omis. Il est prudent d'inclure une esperluette suivie de lettres qui ne forment pas une référence de caractère nommé, mais si les lettres sont remplacées par une chaîne qui *forme* une référence de caractère nommé, elles seront interprétées comme ce caractère à la place.



Dans ce fragment, la valeur de l'attribut est " ?bill&ted" :

```
<a href="?bill&ted">Bill and Ted</a>
```

Dans le fragment suivant, cependant, la valeur de l'attribut est en fait " ?art©", *et non* le " " prévu ?art&copy;, car même sans le point-virgule final, " &copy;" est traité de la même manière que " &copy;"; et est donc interprété comme " ©" :

```
<a href="?art&copy">Art and Copy</a>
```

Pour éviter ce problème, toutes les références de caractères nommés doivent se terminer par un point-virgule, et les utilisations de références de caractères nommés sans point-virgule sont signalées comme des erreurs.

Ainsi, la manière correcte d'exprimer les cas ci-dessus est la suivante:

```
<a href="?bill&ted">Bill and Ted</a> <!-- &ted is ok, since  
it's not a named character reference -->  
<a href="?art&amp;copy;">Art and Copy</a> <!-- the & has to be  
escaped, since &copy; is a named character reference -->
```

## Erreurs impliquant des problèmes d'interopérabilité connus dans les anciens agents utilisateurs

Certaines constructions de syntaxe sont connues pour causer des problèmes particulièrement subtils ou graves dans les agents utilisateurs hérités, et sont donc marquées comme non conformes pour aider les auteurs à les éviter.

Par exemple, c'est pourquoi le caractère U+0060 GRAVE ACCENT (`) n'est pas autorisé dans les attributs sans guillemets. Dans certains agents utilisateurs hérités, il est parfois traité comme un guillemet.

Un autre exemple de ceci est le DOCTYPE, qui est nécessaire pour déclencher [le mode sans bizarrerie](#), car le comportement des agents utilisateurs hérités en [mode bizarrerie](#) est souvent largement non documenté.

## Erreurs qui risquent d'exposer les auteurs à des attaques de sécurité

Certaines restrictions existent uniquement pour éviter les problèmes de sécurité connus.

Par exemple, la restriction sur l'utilisation d'UTF-7 existe uniquement pour éviter que les auteurs ne soient la proie d'une attaque de script intersite connue utilisant UTF-7. [\[UTF7\]](#)

## Cas où l'intention de l'auteur n'est pas claire

Le balisage où l'intention de l'auteur est très peu claire est souvent rendu non conforme. La correction précoce de ces erreurs facilite la maintenance ultérieure.

Par exemple, il n'est pas clair si l'auteur voulait que ce qui suit soit un [h1](#) titre ou un [h2](#) titre :

```
<h1>Contact details</h2>
```

### Cas susceptibles d'être des fautes de frappe

Lorsqu'un utilisateur fait une simple faute de frappe, il est utile que l'erreur puisse être détectée tôt, car cela peut faire gagner beaucoup de temps à l'auteur pour le débogage. Cette spécification considère donc généralement comme une erreur d'utiliser des noms d'éléments, des noms d'attributs, etc., qui ne correspondent pas aux noms définis dans cette spécification.

Par exemple, si l'auteur tapait `<capton>` au lieu de `<caption>`, cela serait signalé comme une erreur et l'auteur pourrait corriger la faute de frappe immédiatement.

### Erreurs susceptibles d'interférer avec la nouvelle syntaxe à l'avenir

Afin de permettre à la syntaxe du langage d'être étendue à l'avenir, certaines fonctionnalités autrement inoffensives sont interdites.

Par exemple, les "attributs" dans les balises de fin sont actuellement ignorés, mais ils ne sont pas valides, au cas où une future modification de la langue utiliserait cette fonctionnalité de syntaxe sans entrer en conflit avec le contenu déjà déployé (et valide !).

Certains auteurs trouvent utile d'avoir l'habitude de toujours citer tous les attributs et d'inclure toujours toutes les balises facultatives, préférant la cohérence dérivée d'une telle coutume aux avantages mineurs de laconisme offerts par l'utilisation de la flexibilité de la syntaxe HTML. Pour aider ces auteurs, les vérificateurs de conformité peuvent fournir des modes de fonctionnement dans lesquels ces conventions sont appliquées.

### 1.11.3 Restrictions sur les modèles de contenu et sur les valeurs d'attribut

*Cette section est non normative.*

Au-delà de la syntaxe du langage, cette spécification impose également des restrictions sur la façon dont les éléments et les attributs peuvent être spécifiés. Ces restrictions sont présentes pour des raisons similaires :

#### Erreurs impliquant du contenu avec une sémantique douteuse

Pour éviter l'utilisation abusive d'éléments avec des significations définies, des modèles de contenu sont définis qui restreignent la manière dont les éléments peuvent être imbriqués lorsque de telles imbrications auraient une valeur douteuse.

Par exemple, cette spécification interdit l'imbrication d'un `section` élément à l'intérieur d'un `kbd` élément, car il est très peu probable qu'un auteur indique qu'une section entière doit être saisie.

## Erreurs impliquant un conflit dans la sémantique exprimée

De même, pour attirer l'attention de l'auteur sur les erreurs d'utilisation des éléments, les contradictions manifestes dans la sémantique exprimée sont également considérées comme des erreurs de conformité.

Dans les fragments ci-dessous, par exemple, la sémantique est absurde : un séparateur ne peut pas être simultanément une cellule, ni un bouton radio une barre de progression.

```
<hr role="cell">  
<input type="radio" role="progressbar">
```

Un autre exemple est les restrictions sur les modèles de contenu de l'ulélément, qui n'autorisent que liles éléments enfants. Par définition, les listes se composent uniquement de zéro ou plusieurs éléments de liste, donc si un ulélément contient autre chose qu'un liélément, ce que cela signifie n'est pas clair.

## Cas où les styles par défaut sont susceptibles de prêter à confusion

Certains éléments ont des styles ou des comportements par défaut qui rendent certaines combinaisons susceptibles de prêter à confusion. Lorsque ceux-ci ont des alternatives équivalentes sans ce problème, les combinaisons déroutantes sont interdites.

Par exemple, divles éléments sont rendus sous forme de boîtes de bloc et spanles éléments sous forme de boîtes en ligne . Mettre une boîte de bloc dans une boîte en ligne est inutilement déroutant ; étant donné que l'imbrication div d'éléments uniquement, ou l'imbrication d'éléments uniquement span, ou l'imbrication spand'éléments à l'intérieur divd'éléments ont tous le même objectif que l'imbrication d'un div élément dans un spanélément, mais seul ce dernier implique une boîte de bloc dans une boîte en ligne , cette dernière combinaison est interdite. Un autre exemple serait la façon dont le contenu interactif ne peut pas être imbriqué. Par exemple, un buttonélément ne peut pas contenir un textarea élément. En effet, le comportement par défaut de ces éléments interactifs imbriqués serait très déroutant pour les utilisateurs. Au lieu d'imbriquer ces éléments, ils peuvent être placés côte à côte.

## Erreurs qui indiquent un malentendu probable de la spécification

Parfois, quelque chose est interdit car l'autoriser entraînerait probablement la confusion de l'auteur.

Par exemple, définir l' disabled attribut sur la valeur " false" n'est pas autorisé, car malgré l'apparence de sens que l'élément est activé, cela signifie en fait que l'élément est désactivé (ce qui compte pour les implémentations est la présence de l'attribut, pas sa valeur ).

## Erreurs impliquant des limites qui ont été imposées simplement pour simplifier le langage

Certaines erreurs de conformité simplifient le langage que les auteurs doivent apprendre.

Par exemple, l'attribut `area` de l'élément `shape`, bien qu'il accepte les deux valeurs `circ` et `circle` dans la pratique comme synonymes, interdit l'utilisation de la `circ` valeur, afin de simplifier les didacticiels et autres aides à l'apprentissage. Il n'y aurait aucun avantage à autoriser les deux, mais cela entraînerait une confusion supplémentaire lors de l'enseignement de la langue.

### Erreurs qui impliquent des particularités de l'analyseur

Certains éléments sont analysés de manière quelque peu excentrique (généralement pour des raisons historiques), et leurs restrictions de modèle de contenu visent à éviter d'exposer l'auteur à ces problèmes.

Par exemple, un `form` élément n'est pas autorisé dans [le phrasing content](#), car lorsqu'il est analysé en tant que HTML, la `form` balise de début d'un élément impliquera la `p` balise de fin d'un élément. Ainsi, le balisage suivant se traduit par deux [paragraphes](#), et non un :

```
<p>Welcome. <form><label>Name:</label> <input></form>
```

Il est analysé exactement comme suit :

```
<p>Welcome. </p><form><label>Name:</label> <input></form>
```

### Erreurs susceptibles d'entraîner l'échec des scripts de manière difficile à déboguer

Certaines erreurs sont destinées à aider à prévenir les problèmes de script qui seraient difficiles à déboguer.

C'est pourquoi, par exemple, il n'est pas conforme d'avoir deux `id` attributs avec la même valeur. Les identifiants en double conduisent à sélectionner le mauvais élément, avec des effets parfois désastreux dont la cause est difficile à déterminer.

### Erreurs qui font perdre du temps à la création

Certaines constructions sont interdites car, historiquement, elles ont été la cause de beaucoup de temps de création perdu, et en encourageant les auteurs à éviter de les créer, les auteurs peuvent gagner du temps dans leurs efforts futurs.

Par exemple, l'attribut `script` d'un élément `src` entraîne l'ignorance du contenu de l'élément. Cependant, ce n'est pas évident, surtout si le contenu de l'élément semble être un script exécutable - ce qui peut amener les auteurs à passer beaucoup de temps à essayer de déboguer le script en ligne sans se rendre compte qu'il ne s'exécute pas. Pour réduire ce problème, cette spécification rend non conforme le fait d'avoir un script exécutable dans un `script` élément lorsque l' `src` attribut est présent. Cela signifie que les auteurs qui valident leurs documents sont moins susceptibles de perdre du temps avec ce genre d'erreur.

## Erreurs impliquant des zones affectant les auteurs migrant entre les syntaxes HTML et XML

Certains auteurs aiment écrire des fichiers qui peuvent être interprétés à la fois comme XML et HTML avec des résultats similaires. Bien que cette pratique soit généralement déconseillée en raison de la myriade de complications subtiles impliquées (en particulier lorsqu'elles impliquent des scripts, des styles ou tout type de sérialisation automatisée), cette spécification comporte quelques restrictions destinées à atténuer au moins quelque peu les difficultés. Cela permet aux auteurs de l'utiliser plus facilement comme étape de transition lors de la migration entre les syntaxes HTML et XML.

Par exemple, il existe des règles quelque peu compliquées entourant les attributs `lang` et `xml:lang` destinés à maintenir les deux synchronisés. Un autre exemple serait les restrictions sur les valeurs des `xmlns` attributs dans la sérialisation HTML, qui visent à garantir que les éléments des documents conformes se retrouvent dans les mêmes espaces de noms, qu'ils soient traités en HTML ou en XML.

## Erreurs impliquant des zones réservées pour une expansion future

Comme pour les restrictions sur la syntaxe destinées à permettre une nouvelle syntaxe dans les futures révisions du langage, certaines restrictions sur les modèles de contenu des éléments et les valeurs des attributs sont destinées à permettre une expansion future du vocabulaire HTML.

Par exemple, limiter les valeurs de l' `target` attribut commençant par un caractère U+005F LOW LINE ( `_` ) à des valeurs prédéfinies spécifiques permet d'introduire ultérieurement de nouvelles valeurs prédéfinies sans entrer en conflit avec les valeurs définies par l'auteur.

## Erreurs indiquant une mauvaise utilisation d'autres spécifications

Certaines restrictions sont destinées à prendre en charge les restrictions imposées par d'autres spécifications.

Par exemple, exiger que les attributs qui acceptent des listes de requêtes multimédias n'utilisent que des listes de requêtes multimédias *valides* renforce l'importance de suivre les règles de conformité de cette spécification.

## 1.12 Lecture suggérée

*Cette section est non normative.*

Les documents suivants pourraient intéresser les lecteurs de cette spécification.

**Modèle de personnage pour le World Wide Web 1.0 : principes de base** [\[CHARMOD\]](#)

*Cette spécification architecturale fournit aux auteurs de spécifications, aux développeurs de logiciels et aux développeurs de contenu une référence commune pour la manipulation de texte interopérable sur le World Wide Web, en s'appuyant sur le jeu de caractères universel, défini conjointement par la norme Unicode et l'ISO/IEC 10646. Les sujets abordés incluent utilisation des termes « caractère », « encodage » et « chaîne », un modèle de traitement de référence, le choix et l'identification des encodages de caractères, l'échappement des caractères et l'indexation des chaînes.*

### **Considérations sur la sécurité Unicode [\[UTR36\]](#)**

*Étant donné qu'Unicode contient un si grand nombre de caractères et intègre les différents systèmes d'écriture du monde, une utilisation incorrecte peut exposer les programmes ou les systèmes à d'éventuelles attaques de sécurité. Ceci est d'autant plus important que de plus en plus de produits sont internationalisés. Ce document décrit certaines des considérations de sécurité que les programmeurs, les analystes système, les développeurs de normes et les utilisateurs doivent prendre en compte, et fournit des recommandations spécifiques pour réduire le risque de problèmes.*

### **Directives pour l'accessibilité du contenu Web (WCAG) [\[WCAG\]](#)**

*Les directives pour l'accessibilité du contenu Web (WCAG) couvrent un large éventail de recommandations pour rendre le contenu Web plus accessible. Le respect de ces directives rendra le contenu accessible à un plus large éventail de personnes handicapées, y compris la cécité et la basse vision, la surdité et la perte auditive, les troubles d'apprentissage, les limitations cognitives, les mouvements limités, les troubles de la parole, la photosensibilité et des combinaisons de ceux-ci. Le respect de ces directives rendra également souvent votre contenu Web plus utilisable pour les utilisateurs en général.*

### **Directives d'accessibilité de l'outil de création (ATAG) 2.0 [\[ATAG\]](#)**

*Cette spécification fournit des lignes directrices pour la conception d'outils de création de contenu Web plus accessibles aux personnes handicapées. Un outil de création conforme à ces directives favorisera l'accessibilité en fournissant une interface utilisateur accessible aux auteurs handicapés ainsi qu'en permettant, soutenant et promouvant la production de contenu Web accessible par tous les auteurs.*

### **Directives d'accessibilité de l'agent utilisateur (UAAG) 2.0 [\[UAAG\]](#)**

*Ce document fournit des lignes directrices pour la conception d'agents utilisateurs qui réduisent les obstacles à l'accessibilité du Web pour les personnes handicapées. Les agents utilisateurs incluent les navigateurs et d'autres types de logiciels qui récupèrent et restituent le contenu Web. Un agent utilisateur qui se conforme à ces directives favorisera l'accessibilité via sa propre interface utilisateur et via d'autres installations internes, y compris sa capacité à communiquer avec d'autres technologies (en particulier les technologies d'assistance). De plus, tous*

*les utilisateurs, et pas seulement les utilisateurs handicapés, devraient trouver des agents utilisateurs conformes plus utilisables.*

## 2 Infrastructures communes

Cette spécification dépend d' *Infra* . [\[INFRA\]](#)

### 2.1 Terminologie

Cette spécification fait référence à la fois aux attributs HTML et XML et aux attributs IDL, souvent dans le même contexte. Lorsqu'il n'est pas clair à quoi il est fait référence, ils sont appelés **attributs de contenu** pour les attributs HTML et XML, et **attributs IDL** pour ceux définis sur les interfaces IDL. De même, le terme "propriétés" est utilisé à la fois pour les propriétés d'objet JavaScript et les propriétés CSS. Lorsque celles-ci sont ambiguës, elles sont respectivement qualifiées de **propriétés d'objet** et de **propriétés CSS** .

Généralement, lorsque la spécification indique qu'une fonctionnalité s'applique à [la syntaxe HTML](#) ou [à la syntaxe XML](#) , elle inclut également l'autre. Lorsqu'une fonctionnalité ne s'applique spécifiquement qu'à l'un des deux langages, elle est appelée en indiquant explicitement qu'elle ne s'applique pas à l'autre format, comme dans "pour HTML, ... (cela ne s'applique pas à XML)".

Cette spécification utilise le terme **document** pour désigner toute utilisation de HTML, allant de courts documents statiques à de longs essais ou rapports multimédias riches, ainsi qu'à des applications interactives à part entière. Le terme est utilisé pour désigner à la fois [Document](#) les objets et leurs arbres DOM descendants, et les flux d'octets sérialisés utilisant la [syntaxe HTML](#) ou la [syntaxe XML](#) , selon le contexte.

Dans le contexte des structures DOM, les termes [document HTML](#) et [document XML](#) sont utilisés tels qu'ils sont définis dans *DOM* , et se réfèrent spécifiquement à deux modes différents dans lesquels [Document](#) les objets peuvent se trouver. [\[DOM\]](#) (De telles utilisations sont toujours liées à leur définition.)

Dans le contexte des flux d'octets, le terme document HTML fait référence aux ressources étiquetées comme [text/html](#), et le terme document XML fait référence aux ressources étiquetées avec un [type MIME XML](#) .

---



Pour plus de simplicité, des termes tels que **affiché** , **affiché** et **visible** peuvent parfois être utilisés pour désigner la manière dont un document est rendu à l'utilisateur. Ces termes ne sont pas censés impliquer un support visuel ; elles doivent être considérées comme s'appliquant à d'autres médias de manière équivalente.

### 2.1.1 Parallélisme

Exécuter des étapes **en parallèle** signifie que ces étapes doivent être exécutées, l'une après l'autre, en même temps que d'autres logiques de la norme (par exemple, en même temps que la [boucle d'événements](#) ). Cette norme ne définit pas le mécanisme précis par lequel cela est réalisé, qu'il s'agisse d'un multitâche coopératif à temps partagé, de fibres, de threads, de processus utilisant différents hyperthreads, cœurs, processeurs, machines, etc. En revanche, une opération qui doit s'exécuter **immédiatement** doit interrompre la tâche en cours d'exécution, s'exécuter, puis reprendre la tâche en cours d'exécution précédente.

*Pour obtenir des conseils sur l'écriture de spécifications qui exploitent le parallélisme, consultez [Gérer la boucle d'événements à partir d'autres spécifications](#) .*

Pour éviter les conditions de concurrence entre différents algorithmes [en parallèle](#) qui fonctionnent sur les mêmes données, une [file d'attente parallèle](#) peut être utilisée.

Une **file d'attente parallèle** représente une file d'attente d'étapes d'algorithme qui doivent être exécutées en série.

Une [file d'attente parallèle](#) a une **file d'attente d'algorithmes** (une [queue](#) ), initialement vide.

Pour **placer des étapes** dans une [file d'attente parallèle](#) , [placez](#) les étapes d'algorithme dans la [file d'attente d'algorithmes](#) de la file d'attente [parallèle](#) .

Pour **démarrer une nouvelle file d'attente parallèle** , exécutez les étapes suivantes :

1. Soit *parallelQueue* une nouvelle [file d'attente parallèle](#) .
2. Exécutez les étapes suivantes [en parallèle](#) :
  1. Alors que c'est vrai :
    1. Supposons que *les étapes* soient le résultat du [retrait](#) de la file d'attente de l' [algorithme](#) de *parallelQueue* .
    2. Si *étapes* n'est pas rien, exécutez *étapes* .



3. [Assert](#) : les étapes en cours d'exécution n'ont pas levé d'exception, car les étapes s'exécutant [en parallèle](#) ne sont pas autorisées à lever.

*Les implémentations ne sont pas censées implémenter cela comme une boucle fonctionnant en continu. Les algorithmes des normes doivent être faciles à comprendre et ne sont pas nécessairement bons pour la durée de vie ou les performances de la batterie.*

### 3. Retourne `parallelQueue` .

*Les étapes s'exécutant [en parallèle](#) peuvent elles-mêmes exécuter d'autres étapes [en parallèle](#) . Par exemple, à l'intérieur d'une [file d'attente parallèle](#) , il peut être utile d'exécuter une série d'étapes [en parallèle](#) avec la file d'attente.*

*Imaginez une `nameList` standard définie (une [list](#) ), ainsi qu'une méthode pour ajouter un `nom` à `nameList` , à moins que `nameList` [ne contienne](#) déjà `name` , auquel cas elle rejette.*

La solution suivante souffre de conditions de concurrence :

1. Soit `p` une nouvelle promesse.
2. Exécutez les étapes suivantes [en parallèle](#) :
  1. Si `nameList` [contient](#) `name` , rejetez `p` avec a `TypeError` et abandonnez ces étapes.
  2. Faites des travaux potentiellement longs.
  3. [Ajouter](#) `name` à `nameList` .
  4. Résoudre `p` avec indéfini.
3. Retour `p` .

Deux invocations de ce qui précède peuvent s'exécuter simultanément, ce qui signifie que `name` n'est pas dans `nameList` lors de l'étape 2.1, mais il *peut être ajouté* avant l'exécution de l'étape 2.3, ce qui signifie que `name` se retrouve deux fois dans `nameList` .

Les files d'attente parallèles résolvent ce problème. La norme laisserait `nameListQueue` être le résultat du [démarrage d'une nouvelle file d'attente parallèle](#) , puis :

1. Soit `p` une nouvelle promesse.
2. [Mettez les étapes suivantes en file d'attente](#) dans `nameListQueue` :
  1. Si `nameList` [contient](#) `name` , rejetez `p` avec a `TypeError` et abandonnez ces étapes.

2. Faites des travaux potentiellement longs.
3. [Ajouter](#) *name* à *nameList*.
4. Résoudre *p* avec indéfini.

3. Retour *p*.

Les marches feraient désormais la queue et la course serait évitée.

### 2.1.2 Ressources

La spécification utilise le terme **pris en charge** lorsqu'il s'agit de savoir si un agent utilisateur a une implémentation capable de décoder la sémantique d'une ressource externe. Un format ou un type est dit *pris en charge* si l'implémentation peut traiter une ressource externe de ce format ou de ce type sans que les aspects critiques de la ressource soient ignorés. *La prise en charge* d'une ressource spécifique peut dépendre des fonctionnalités du format de la ressource qui sont utilisées.

Par exemple, une image PNG serait considérée comme étant dans un format pris en charge si ses données de pixel pouvaient être décodées et restituées, même si, à l'insu de l'implémentation, l'image contenait également des données d'animation. Un fichier vidéo MPEG-4 ne serait pas considéré comme étant dans un format pris en charge si le format de compression utilisé n'était pas pris en charge, même si l'implémentation pouvait déterminer les dimensions du film à partir des métadonnées du fichier.

Ce que certaines spécifications, en particulier les spécifications HTTP, appellent une *représentation* est appelé dans cette spécification une **ressource**. [\[HTTP\]](#)

**Les sous-ressources critiques** d'une ressource sont celles dont la ressource doit disposer pour être correctement traitées. Les ressources considérées comme critiques ou non sont définies par la spécification qui définit le format de la ressource.

Pour [les feuilles de style CSS](#), nous définissons provisoirement ici que leurs sous-ressources critiques sont d'autres feuilles de style importées via `@import` des règles, y compris celles importées indirectement par d'autres feuilles de style importées.

Cette définition n'est pas totalement interopérable ; en outre, certains agents utilisateurs semblent considérer des ressources telles que les images d'arrière-plan ou les polices Web comme des sous-ressources critiques. Idéalement, le groupe de travail CSS définirait cela ; voir [w3c/csswg-drafts numéro 1088](#) pour suivre les progrès sur ce front.

### 2.1.3 Compatibilité XML

Pour faciliter la migration de HTML vers XML, les agents utilisateurs conformes à cette spécification placeront des éléments en HTML dans l' <http://www.w3.org/1999/xhtml> espace de noms, au moins pour les besoins du DOM et du CSS. Le terme « **éléments HTML** » fait référence à tout élément de cet espace de noms, même dans les documents XML.

Sauf indication contraire, tous les éléments définis ou mentionnés dans cette spécification sont dans l' [espace de noms HTML](http://www.w3.org/1999/xhtml) (" <http://www.w3.org/1999/xhtml>"), et tous les attributs définis ou mentionnés dans cette spécification n'ont pas d'espace de noms.

Le terme **type d'élément** est utilisé pour désigner l'ensemble d'éléments qui ont un nom local et un espace de noms donnés. Par exemple, `button` les éléments sont des éléments avec le type d'élément `button`, ce qui signifie qu'ils ont le nom local " `button`" et (implicitement comme défini ci-dessus) l' [espace de noms HTML](http://www.w3.org/1999/xhtml) .

Les noms d'attributs sont dits **compatibles XML** s'ils correspondent à la `Name` production définie en XML et s'ils ne contiennent pas de caractères U+003A COLON (:). [\[XML\]](#)

### 2.1.4 Arbres DOM

Lorsqu'il est indiqué qu'un élément ou un attribut est **ignoré** , ou traité comme une autre valeur, ou traité comme s'il s'agissait d'autre chose, cela se réfère uniquement au traitement du nœud une fois qu'il est dans le DOM. Un agent utilisateur ne doit pas muter le DOM dans de telles situations.

On dit qu'un attribut de contenu ne **change** de valeur que si sa nouvelle valeur est différente de sa valeur précédente ; définir un attribut sur une valeur qu'il a déjà ne le modifie pas.

Le terme **empty** , lorsqu'il est utilisé pour une valeur d'attribut, `Text` un nœud ou une chaîne, signifie que la [longueur](#) du texte est nulle (c'est-à-dire qu'il ne contient même pas [de contrôles](#) ou U+0020 SPACE).

Un élément HTML peut avoir **des étapes d'insertion d'élément HTML** spécifiques définies pour le [nom local](#) de l'élément . De même, un élément HTML peut avoir **des étapes de suppression d'élément HTML** spécifiques définies pour le [nom local](#) de l'élément .

Les [étapes d'insertion](#) pour le standard HTML, étant donné *insertNode* , sont définies comme suit :

1. Si *insertNode* est un élément dont [l'espace de noms](#) est l' [espace de noms HTML](#) , et que cette norme définit [les étapes d'insertion d'éléments HTML](#) pour [le nom local](#) d' *insertNode* , alors exécutez les [étapes d'insertion d'éléments HTML](#) correspondantes étant donné *insertNode* .
2. Si *insertNode* est un [élément associé au formulaire](#) ou l'ancêtre d'un [élément associé au formulaire](#) , alors :
  1. Si l' [indicateur d'insertion de l'analyseur](#) de [l'élément associé au formulaire](#) est défini, alors retournez.
  2. [Réinitialisez le propriétaire du formulaire](#) de l' [élément associé au formulaire](#) .

Les [étapes de suppression](#) pour le standard HTML, données *removeNode* et éventuellement *oldParent* , sont définies comme suit :

1. Si [la zone focalisée](#) du document du [noeud](#) de *removeNode* est *removeNode* , alors définissez [la zone focalisée](#) du document du [noeud](#) de *removeNode* sur [la fenêtre d'affichage](#) du document du [noeud](#) de *removeNode* .

*Cela n'exécute pas les [étapes de non mise au point](#) , les [étapes de mise au point](#) ou les [étapes de mise à jour de la mise au point](#) , et donc aucun événement `blur` ou `change` n'est déclenché.*

2. Si *removeNode* est un élément dont [l'espace de noms](#) est l' [espace de noms HTML](#) , et que ce standard définit [les étapes de suppression d'éléments HTML](#) pour [le nom local](#) de *disabledNode* , alors exécutez les [étapes de suppression d'éléments HTML](#) correspondantes étant donné *removeNode* et éventuellement *oldParent* .
3. Si *removeNode* est un [élément associé au formulaire](#) ou l'ancêtre d'un [élément associé au formulaire](#) , alors :
  1. Si l' [élément associé au formulaire](#) a un [propriétaire de formulaire](#) et que l' [élément associé au formulaire](#) et son [propriétaire de formulaire](#) ne se trouvent plus dans la même [arborescence](#) , [réinitialisez le propriétaire de formulaire](#) de l' [élément associé au formulaire](#) .
4. Si l'attribut de *removeNode* `popover` n'est pas dans l' [état no popover](#) , exécutez alors l' [algorithme de masquage du popover](#) en indiquant *removeNode* , false, false et false.

Un **nœud est inséré dans un document** lorsque les [étapes d'insertion](#) sont appelées avec lui comme argument et il se trouve maintenant [dans une arborescence de documents](#) . De manière analogue, un **nœud est supprimé d'un document** lorsque les [étapes de suppression](#) sont invoquées avec lui comme argument et il n'est plus [dans une arborescence de documents](#) .

Un nœud **devient connecté** lorsque les [étapes d'insertion](#) sont appelées avec lui comme argument et il est maintenant [connecté](#) . De manière analogue, un nœud **devient déconnecté** lorsque les [étapes de suppression](#) sont appelées avec lui comme argument et il n'est plus [connecté](#) .

Un nœud est **connecté au contexte de navigation** lorsqu'il est [connecté](#) et que [le contexte de navigation de son ombre, y compris la racine](#), n'est pas nul. Un nœud **devient connecté au contexte de navigation** lorsque les [étapes d'insertion](#) sont invoquées avec lui comme argument et il est maintenant [connecté au contexte de navigation](#) . Un nœud **devient déconnecté du contexte de navigation** soit lorsque les [étapes de suppression](#) sont invoquées avec lui comme argument et qu'il n'est plus [connecté au contexte de navigation](#) , soit lorsque [le contexte de navigation](#) de son ombre, y compris la [racine](#), devient nul.

### 2.1.5 Script

La construction "un FooObjet", où Foo est en fait une interface, est parfois utilisée à la place du plus précis "un objet implémentant l'interface Foo".

On dit qu'un attribut IDL est en cours **d'obtention** lorsque sa valeur est récupérée (par exemple par le script de l'auteur), et qu'il est **défini** lorsqu'une nouvelle valeur lui est assignée.

Si un objet DOM est dit **live** , les attributs et les méthodes de cet objet doivent fonctionner sur les données sous-jacentes réelles, et non sur un instantané des données.

### 2.1.6 Plugins

Le terme **plug-in** fait référence à un ensemble [défini par l'implémentation](#) de gestionnaires de contenu utilisés par l'agent utilisateur qui peuvent prendre part au rendu d'un Document objet par l'agent utilisateur, mais qui n'agissent pas en tant [qu'enfants navigables](#) de Document ni n'introduisent d' Node objets dans le Document DOM de .

Généralement, ces gestionnaires de contenu sont fournis par des tiers, bien qu'un agent utilisateur puisse également désigner des gestionnaires de contenu intégrés en tant que plug-ins.

Un agent utilisateur ne doit pas considérer les types `text/plain` et `application/octet-stream` comme ayant un [plugin](#) enregistré .

Un exemple de plugin serait un visualiseur PDF qui est instancié dans un [navigable](#) lorsque l'utilisateur navigue vers un fichier PDF. Cela compterait comme un plug-in, que la partie qui a implémenté le composant de visionneuse PDF soit ou non la même que celle qui a implémenté l'agent utilisateur lui-même. Cependant, une application de visualisation de PDF qui se lance séparément de l'agent utilisateur (par opposition à l'utilisation de la même interface) n'est pas un plug-in selon cette définition.

*Cette spécification ne définit pas de mécanisme pour interagir avec les plugins, car il est censé être spécifique à l'agent utilisateur et à la plate-forme. Certains agents utilisateur peuvent choisir de prendre en charge un mécanisme de plug-in tel que l'API de plug-in Netscape ; d'autres peuvent utiliser des convertisseurs de contenu à distance ou avoir un support intégré pour certains types. En effet, cette spécification n'exige pas du tout que les agents utilisateurs prennent en charge les plugins. [\[NPAPI\]](#)*

**Les navigateurs doivent être extrêmement prudents lorsqu'ils interagissent avec du contenu externe destiné aux [plugins](#) . Lorsque le logiciel tiers est exécuté avec les mêmes privilèges que l'agent utilisateur lui-même, les vulnérabilités du logiciel tiers deviennent aussi dangereuses que celles de l'agent utilisateur.**

Étant donné que différents utilisateurs ayant différents ensembles de [plugins](#) fournissent un vecteur de suivi qui augmente les chances que les utilisateurs soient identifiés de manière unique, les agents utilisateurs sont encouragés à prendre en charge exactement le même ensemble de [plugins](#) pour chaque utilisateur.

### 2.1.7 Codages des caractères

Un [codage de caractères](#) , ou simplement *un codage* où cela n'est pas ambigu, est un moyen défini de convertir entre les flux d'octets et les chaînes Unicode, comme défini dans *Encoding* . Un [encodage](#) a un [nom d'encodage](#) et une ou plusieurs [étiquettes d'encodage](#) , appelées *nom* et *étiquettes* d'encodage dans la norme d'encodage. [\[CODAGE\]](#)

### 2.1.8 Classes de conformité

Cette spécification décrit les critères de conformité pour les agents utilisateurs (pertinents pour les implémenteurs) et les documents (pertinents pour les auteurs et les implémenteurs d'outils de création).

**Les documents conformes** sont ceux qui satisfont à tous les critères de conformité des documents. Pour des raisons de lisibilité, certaines de ces exigences de conformité sont formulées comme des exigences de conformité imposées aux auteurs ; ces exigences sont implicitement des exigences sur les documents : par définition, tous les documents sont supposés avoir eu un auteur. (Dans certains cas, cet auteur peut lui-même être un agent utilisateur — ces agents utilisateurs sont soumis à des règles supplémentaires, comme expliqué ci-dessous.)

Par exemple, si une exigence stipule que "les auteurs ne doivent pas utiliser l' `foobar`élément", cela impliquerait que les documents ne sont pas autorisés à contenir des éléments nommés `foobar`.

*Il n'y a pas de relation implicite entre les exigences de conformité de document et les exigences de conformité d'implémentation. Les agents utilisateurs ne sont pas libres de traiter les documents non conformes comme bon leur semble ; le modèle de traitement décrit dans cette spécification s'applique aux implémentations indépendamment de la conformité des documents d'entrée.*

Les agents utilisateurs appartiennent à plusieurs catégories (qui se chevauchent) avec différentes exigences de conformité.

### **Navigateurs Web et autres agents utilisateurs interactifs**

Les navigateurs Web qui prennent en charge [la syntaxe XML](#) doivent traiter les éléments et les attributs de l' [espace de noms HTML](#) trouvé dans les documents XML comme décrit dans cette spécification, afin que les utilisateurs puissent interagir avec eux, à moins que la sémantique de ces éléments n'ait été remplacée par d'autres spécifications.

Un navigateur Web conforme, lorsqu'il trouve un `script`élément dans un document XML, exécute le script contenu dans cet élément. Cependant, si l'élément est trouvé dans une transformation exprimée en XSLT (en supposant que l'agent utilisateur supporte également XSLT), le processeur traitera plutôt l' `script`élément comme un élément opaque qui fait partie de la transformation.

Les navigateurs Web qui prennent en charge [la syntaxe HTML](#) doivent traiter les documents étiquetés avec un [type HTML MIME](#) comme décrit dans cette spécification, afin que les utilisateurs puissent interagir avec eux.

Les agents utilisateurs qui prennent en charge les scripts doivent également être des implémentations conformes des fragments IDL de cette spécification, comme décrit dans *Web IDL* . [\[WEBIDL\]](#)

*Sauf indication explicite, les spécifications qui remplacent la sémantique des éléments HTML ne remplacent pas les exigences sur les objets DOM représentant ces éléments. Par exemple, l' `script`élément de l'exemple ci-dessus implémenterait toujours l' `HTMLScriptElement` interface.*

### **Agents utilisateurs de présentation non interactifs**

Les agents utilisateurs qui traitent des documents HTML et XML uniquement pour en rendre des versions non interactives doivent se conformer aux mêmes



critères de conformité que les navigateurs Web, sauf qu'ils sont exemptés des exigences concernant l'interaction de l'utilisateur.

*Des exemples typiques d'agents utilisateurs de présentation non interactifs sont les imprimantes (UA statiques) et les écrans suspendus (UA dynamiques). On s'attend à ce que la plupart des agents utilisateurs de présentation non interactifs statiques choisissent également de ne pas prendre en charge les scripts .*

Un UA de présentation non interactif mais dynamique exécuterait toujours des scripts, permettant aux formulaires d'être soumis dynamiquement, et ainsi de suite. Cependant, étant donné que le concept de "focus" n'est pas pertinent lorsque l'utilisateur ne peut pas interagir avec le document, l'UA n'aurait pas besoin de prendre en charge l'une des API DOM liées au focus.

### **Agents utilisateurs visuels prenant en charge le rendu par défaut suggéré**

Les agents utilisateurs, qu'ils soient interactifs ou non, peuvent être désignés (éventuellement en tant qu'option utilisateur) comme prenant en charge le rendu par défaut suggéré défini par la présente spécification.

Ce n'est pas obligatoire. En particulier, même les agents utilisateurs qui implémentent le rendu par défaut suggéré sont encouragés à proposer des paramètres qui remplacent cette valeur par défaut pour améliorer l'expérience de l'utilisateur, par exemple en modifiant le contraste des couleurs, en utilisant différents styles de focus ou en rendant l'expérience plus accessible et utilisable. à l'utilisateur.

Les agents utilisateurs qui sont désignés comme prenant en charge le rendu par défaut suggéré doivent, pendant qu'ils sont ainsi désignés, implémenter les règles que [la section Rendu](#) définit comme le comportement que les agents utilisateurs sont censés implémenter.

### **Agents utilisateurs sans prise en charge des scripts**

Les implémentations qui ne prennent pas en charge les scripts (ou dont les fonctionnalités de script sont entièrement désactivées) sont exemptées de la prise en charge des événements et des interfaces DOM mentionnés dans cette spécification. Pour les parties de cette spécification qui sont définies en termes de modèle d'événements ou en termes de DOM, ces agents utilisateurs doivent toujours agir comme si les événements et le DOM étaient pris en charge.

*Les scripts peuvent faire partie intégrante d'une application. Les navigateurs Web qui ne prennent pas en charge les scripts, ou dont les scripts sont désactivés, peuvent ne pas être en mesure de transmettre pleinement l'intention de l'auteur.*

### **Vérificateurs de conformité**

Les vérificateurs de conformité doivent vérifier qu'un document est conforme aux critères de conformité applicables décrits dans la présente spécification. Les vérificateurs de conformité automatisés sont dispensés de détecter les erreurs qui nécessitent une interprétation de l'intention de l'auteur



(par exemple, alors qu'un document est non conforme si le contenu d'un élément n'est pas une citation, les vérificateurs de conformité `blockquote` exécutés sans l'intervention d'un jugement humain n'ont pas à vérifier que `blockquote` les éléments ne contiennent que du matériel cité).

Les vérificateurs de conformité doivent vérifier que le document d'entrée est conforme lorsqu'il est analysé sans [contexte de navigation](#) (ce qui signifie qu'aucun script n'est exécuté et que l'[indicateur de script](#) de l'analyseur est désactivé), et doivent également vérifier que le document d'entrée est conforme lorsqu'il est analysé avec un [contexte de navigation](#) dans lequel les scripts s'exécutent et que les scripts ne provoquent jamais d'états non conformes autrement que de manière transitoire pendant l'exécution du script lui-même. (Il ne s'agit que d'une exigence "DEVRAIT" et non d'une exigence "DOIT" car il s'est avéré impossible. [\[CALCUTABLE\]](#) )

Le terme "validateur HTML" peut être utilisé pour désigner un vérificateur de conformité qui se conforme lui-même aux exigences applicables de la présente spécification.

*Les DTD XML ne peuvent pas exprimer toutes les exigences de conformité de cette spécification. Par conséquent, un processeur XML validant et une DTD ne peuvent pas constituer un vérificateur de conformité. De plus, étant donné qu'aucun des deux formats de création définis dans cette spécification n'est une application de SGML, un système de validation SGML ne peut pas non plus constituer un vérificateur de conformité.*

*Autrement dit, il existe trois types de critères de conformité :*

- 1. Critères pouvant être exprimés dans une DTD.*
- 2. Critères qui ne peuvent pas être exprimés par une DTD, mais qui peuvent toujours être vérifiés par une machine.*
- 3. Critères qui ne peuvent être vérifiés que par un humain.*

*Un vérificateur de conformité doit vérifier les deux premiers. Un simple validateur basé sur DTD ne vérifie que la première classe d'erreurs et n'est donc pas un vérificateur de conformité conforme selon cette spécification.*

## **Outils d'exploration de données**

Les applications et les outils qui traitent des documents HTML et XML pour des raisons autres que le rendu des documents ou leur vérification de conformité doivent agir conformément à la sémantique des documents qu'ils traitent.

Un outil qui génère [des plans de document](#) mais augmente le niveau d'imbrication de chaque paragraphe et n'augmente pas le niveau d'imbrication des [titres](#) ne serait pas conforme.

## **Outils de création et générateurs de balisage**

Les outils de création et les générateurs de balisage doivent générer [des documents conformes](#) . Les critères de conformité qui s'appliquent aux auteurs s'appliquent également aux outils de création, le cas échéant.

Les outils de création sont exemptés des exigences strictes d'utilisation des éléments uniquement dans le but spécifié, mais uniquement dans la mesure où les outils de création ne sont pas encore en mesure de déterminer l'intention de l'auteur. Cependant, les outils auteurs ne doivent pas automatiquement abuser des éléments ou inciter leurs utilisateurs à le faire.

Par exemple, il n'est pas conforme d'utiliser un `address` élément pour des informations de contact arbitraires ; cet élément ne peut être utilisé que pour marquer les informations de contact de son ancêtre le plus proche `article` ou `body` de l'élément. Cependant, étant donné qu'un outil de création est probablement incapable de déterminer la différence, un outil de création est exempté de cette exigence. Cela ne signifie pas, cependant, que les outils de création peuvent utiliser `address` des éléments pour n'importe quel bloc de texte en italique (par exemple) ; cela signifie simplement que l'outil de création n'a pas à vérifier que lorsque l'utilisateur utilise un outil pour insérer des informations de contact pour un `article` élément, que l'utilisateur fait vraiment cela et n'insère pas autre chose à la place.

*En termes de vérification de conformité, un éditeur doit produire des documents conformes dans la même mesure qu'un vérificateur de conformité vérifiera.*

Lorsqu'un outil d'édition est utilisé pour éditer un document non conforme, il peut conserver les erreurs de conformité dans les sections du document qui n'ont pas été éditées pendant la session d'édition (c'est-à-dire qu'un outil d'édition est autorisé à aller-retour avec un contenu erroné). Cependant, un outil auteur ne doit pas prétendre que la sortie est conforme si les erreurs ont été ainsi préservées.

Les outils de création devraient se décliner en deux grandes variétés : les outils qui fonctionnent à partir de données structurales ou sémantiques, et les outils qui fonctionnent sur une base d'édition spécifique au média What-You-See-Is-What-You-Get (WYSIWYG).

Le premier est le mécanisme préféré pour les outils qui créent du HTML, car la structure des informations source peut être utilisée pour faire des choix éclairés concernant les éléments et les attributs HTML les plus appropriés.

Cependant, les outils WYSIWYG sont légitimes. Les outils WYSIWYG doivent utiliser des éléments dont ils savent qu'ils sont appropriés et ne doivent pas utiliser d'éléments dont ils ne savent pas qu'ils sont appropriés. Cela peut signifier, dans certains cas extrêmes, limiter l'utilisation des éléments de flux à quelques éléments, tels que `div`, `b`, `i`, et `span` et faire un usage libéral de l' `style` attribut.

Tous les outils de création, qu'ils soient WYSIWYG ou non, doivent s'efforcer de permettre aux utilisateurs de créer un contenu bien structuré, sémantiquement riche et indépendant des médias.

Pour assurer la compatibilité avec le contenu existant et les spécifications antérieures, cette spécification décrit deux formats de création : l'un basé sur [XML](#) et l'autre utilisant un [format personnalisé](#) inspiré de SGML (appelé [syntaxe HTML](#) ). Les implémentations doivent prendre en charge au moins un de ces deux formats, bien que la prise en charge des deux soit encouragée.

Certaines exigences de conformité sont exprimées en tant qu'exigences portant sur des éléments, des attributs, des méthodes ou des objets. De telles exigences entrent dans deux catégories : celles décrivant les restrictions du modèle de contenu et celles décrivant le comportement de mise en œuvre. Ceux de la première catégorie sont des exigences sur les documents et les outils de création. Ceux de la deuxième catégorie sont des exigences sur les agents utilisateurs. De même, certaines exigences de conformité sont formulées comme des exigences imposées aux auteurs ; ces exigences doivent être interprétées comme des exigences de conformité sur les documents produits par les auteurs. (En d'autres termes, cette spécification ne fait pas de distinction entre les critères de conformité sur les auteurs et les critères de conformité sur les documents.)

## 2.1.9 Dépendances

Cette spécification repose sur plusieurs autres spécifications sous-jacentes.

### Infra

Les termes suivants sont définis dans *Infra* : [\[INFRA\]](#)

- Les termes d'itération généraux [while](#) , [continue](#) et [break](#) .
- [Affirmer](#)
- [défini par la mise en œuvre](#)
- [vecteur de suivi](#)
- [point de code](#) et son [caractère synonyme](#)
- [substitut](#)
- [valeur scalaire](#)
- [tuple](#)
- [non-caractère](#)
- [chaîne](#) , [unité de code](#) , [préfixe d'unité de code](#) , [unité de code inférieure à](#) , [commence par](#) , [se termine par](#) , [longueur](#) et [longueur du point de code](#)
- Les opérations d'égalité de chaîne [sont](#) et [identiques à](#)
- [chaîne de valeur scalaire](#)
- [convertir](#)
- [Chaîne ASCII](#)
- [Espace blanc ASCII](#)
- [contrôle](#)
- [Chiffre ASCII](#)
- [Chiffre hexadécimal supérieur ASCII](#)
- [Chiffre hexadécimal inférieur ASCII](#)
- [Chiffre hexadécimal ASCII](#)

- [Alpha supérieur ASCII](#)
- [Alpha inférieur ASCII](#)
- [Alpha ASCII](#)
- [ASCII alphanumérique](#)
- [décodage isomorphe](#)
- [encodage isomorphe](#)
- [ASCII minuscule](#)
- [ASCII majuscule](#)
- [ASCII insensible à la casse](#)
- [supprimer les retours à la ligne](#)
- [normaliser les retours à la ligne](#)
- [supprimer les espaces blancs ASCII de début et de fin](#)
- [supprimer et réduire les espaces blancs ASCII](#)
- [diviser une chaîne sur un espace blanc ASCII](#)
- [diviser une chaîne par des virgules](#)
- [collecter une séquence de points de code](#) et sa [variable de position associée](#)
- [ignorer les espaces blancs ASCII](#)
- La structure de données de [carte ordonnée et les définitions associées pour clé](#) , [valeur](#) , [vide](#) , [entrée](#) , [existent](#) , [obtenir la valeur d'une entrée](#) , [définir la valeur d'une entrée](#) , [supprimer une entrée](#) , [effacer](#) , [obtenir les clés](#) , [obtenir les valeurs](#) , [trier ordre décroissant](#) , [taille](#) et [itération](#)
- La structure de données [de la liste et les définitions associées pour ajouter](#) , [étendre](#) , [préfixer](#) , [remplacer](#) , [supprimer](#) , [vide](#) , [contient](#) , [taille](#) , [indices](#) , [est vide](#) , [élément](#) , [itérer](#) et [clone trier par ordre croissant trier par ordre décroissant](#)
- La structure de données [de la pile](#) et les définitions associées pour [push](#) et [pop](#)
- La structure des données [de la file d'attente](#) et les définitions associées pour [la mise en file d'attente](#) et [le retrait de la file d'attente](#)
- La structure de données [de l'ensemble ordonné](#) et la définition associée pour [l'ajout](#) et [l'union](#)
- Le type de spécification [de structure](#) et la définition associée pour [l'élément](#)
- La structure de données [de la séquence d'octets](#)
- Les algorithmes d'encodage [pardonnant-base64](#) et [de décodage pardonnant-base64](#)
- [gamme exclusive](#)
- [analyser une chaîne JSON en une valeur Infra](#)
- [Espace de noms HTML](#)
- [Espace de noms MathML](#)
- [Espace de noms SVG](#)
- [Espace de noms XLink](#)
- [Espace de noms XML](#)
- [Espace de noms XMLNS](#)

Unicode et encodage

Le jeu de caractères Unicode est utilisé pour représenter les données textuelles et *Encoding* définit les exigences relatives [aux encodages de caractères](#) . [\[UNICODE\]](#)

*Cette spécification [introduit une terminologie](#) basée sur les termes définis dans ces spécifications, comme décrit précédemment.*

Les termes suivants sont utilisés tels que définis dans *Encoding* : [\[ENCODAGE\]](#)

- [Obtenir un encodage](#)
- [Obtenir un encodage de sortie](#)
- [L'algorithme de décodage](#) générique qui prend un flux d'octets et un encodage et renvoie un flux de caractères
- L' algorithme [de décodage UTF-8](#) qui prend un flux d'octets et renvoie un flux de caractères, supprimant en outre une marque d'ordre d'octet (BOM) UTF-8 en tête, le cas échéant
- Le [décodage UTF-8 sans algorithme BOM](#) qui est identique au [décodage UTF-8](#) sauf qu'il ne supprime pas une marque d'ordre d'octet UTF-8 (BOM) en tête
- L' algorithme [d'encodage](#) qui prend un flux de caractères et un encodage et renvoie un flux d'octets
- L' algorithme [d'encodage UTF-8](#) qui prend un flux de caractères et renvoie un flux d'octets
- L' algorithme [de détection de BOM](#) qui prend un flux d'octets et renvoie un codage ou null.

## XML et spécifications associées

Les implémentations qui prennent en charge [la syntaxe XML](#) pour HTML doivent prendre en charge une version de XML, ainsi que sa spécification d'espaces de noms correspondante, car cette syntaxe utilise une sérialisation XML avec des espaces de noms. [\[XML\]](#) [\[XMLNS\]](#)

Les outils d'exploration de données et autres agents utilisateurs qui effectuent des opérations sur le contenu sans exécuter de scripts, évaluer des expressions CSS ou XPath, ou autrement exposer le DOM résultant à un contenu arbitraire, peuvent "prendre en charge les espaces de noms" en affirmant simplement que leurs analogues de nœud DOM se trouvent dans certains espaces de noms, sans exposer réellement les chaînes d'espace de noms.

*Dans [la syntaxe HTML](#) , les préfixes d'espace de noms et les déclarations d'espace de noms n'ont pas le même effet qu'en XML. Par exemple, les deux-points n'ont pas de signification particulière dans les noms d'éléments HTML.*

---

L'attribut avec le nom space dans l' espace de noms XML est défini par *Extensible Markup Language* ( XML ). [\[XML\]](#)

La Name production est définie en XML . [\[XML\]](#)

Cette spécification fait également référence à <?xml-stylesheet?> l'instruction de traitement, définie dans *Associer des feuilles de style à des documents XML* . [\[XMLSSPI\]](#)

Cette spécification mentionne également de manière non normative l' XSLTProcessor interface et ses transformToFragment() méthodes transformToDocument() . [\[XSLTP\]](#)

## URL

Les termes suivants sont définis dans l'*URL* : [\[URL\]](#)

- [héberger](#)
- [suffixe public](#)
- [domaine](#)
- [adresse IP](#)
- [URL](#)
- [Origine](#) des URL
- [URL absolue](#)
- [URL relative](#)
- [domaine enregistrable](#)
- L' [analyseur d'URL](#)
- L' [analyseur d'URL de base](#) et ses arguments de remplacement [d'url](#) et [d'état](#) , ainsi que ces états d'analyseur :
  - [état de début du schéma](#)
  - [état hôte](#)
  - [état du nom d'hôte](#)
  - [état du port](#)
  - [état de début de chemin](#)
  - [état de la requête](#)
  - [état fragmentaire](#)
- [Enregistrement d'URL](#) , ainsi que ses composants individuels :
  - [schème](#)
  - [nom d'utilisateur](#)
  - [mot de passe](#)
  - [héberger](#)
  - [port](#)
  - [chemin](#)
  - [mettre en doute](#)
  - [fragment](#)
  - [entrée d'URL blob](#)
- [chaîne d'URL valide](#)
- Le [ne peut pas avoir de](#) concept nom d'utilisateur/mot de passe/port
- Le concept [de chemin opaque](#)
- [Sérialiseur d'URL](#) et son argument [d'exclusion de fragment](#)

- [Sérialiseur de chemin d'URL](#)
- L' [analyseur hôte](#)
- Le [sérialiseur d'hôte](#)
- [L'hôte est égal à](#)
- [L'URL est égale à](#) et son argument [d'exclusion des fragments](#)
- [sérialiser un entier](#)
- [Jeu de codage par défaut](#)
- [jeu de codage en pourcentage des composants](#)
- [Encodage en pourcentage UTF-8](#)
- [pourcentage de décodage](#)
- [définir le nom d'utilisateur](#)
- [définir le mot de passe](#)
- La [application/x-www-form-urlencoded](#) formule
- Le [application/x-www-form-urlencoded](#) [sérialiseur](#)
- [est spécial](#)

Un certain nombre de schémas et de protocoles sont également référencés par cette spécification :

- Le [about:](#) régime [\[À PROPOS\]](#)
- Le [blob:](#) schéma [\[FILEAPI\]](#)
- Le [data:](#) schéma [\[RFC2397\]](#)
- Le [http:](#) schéma [\[HTTP\]](#)
- Le [https:](#) schéma [\[HTTP\]](#)
- Le [mailto:](#) schéma [\[MAILTO\]](#)
- Le [sms:](#) schéma [\[SMS\]](#)
- Le [urn:](#) schéma [\[URN\]](#)

[La syntaxe des fragments de média](#) est définie dans *l'URI des fragments de média* . [\[MÉDIAFRAG\]](#)

## HTTP et spécifications associées

Les termes suivants sont définis dans les spécifications HTTP : [\[HTTP\]](#)

- [`Accept`](#) en-tête
- [`Accept-Language`](#) en-tête
- [`Cache-Control`](#) en-tête
- [`Content-Disposition`](#) en-tête
- [`Content-Language`](#) en-tête
- [`Content-Range`](#) en-tête
- [`Last-Modified`](#) en-tête
- [`Range`](#) en-tête
- [`Referer`](#) en-tête

Les termes suivants sont définis dans *HTTP State Management Mechanism* : [\[COOKIES\]](#)

- [chaîne de cookies](#)
- [reçoit une chaîne set-cookie](#)
- [`Cookie`](#) en-tête

Le terme suivant est défini dans *Web Linking* : [\[WEBLINK\]](#)



- [`Link`](#) en-tête
- [Analyser une \[Link\]\(#\) valeur de champ ``](#)

Les termes suivants sont définis dans *les valeurs de champ structuré pour HTTP* : [\[STRUCTURED-FIELDS\]](#)

- [en-tête structuré](#)
- [booléen](#)
- [jeton](#)
- [paramètres](#)

Les termes suivants sont définis dans *MIME Sniffing* : [\[MIMESNIFF\]](#)

- [Type MIME](#)
- [Essence de type MIME](#)
- [chaîne de type MIME valide](#)
- [chaîne de type MIME valide sans paramètres](#)
- [Type MIME HTML](#)
- [Le type JavaScript MIME](#) et [l'essence du type JavaScript MIME correspondent](#)
- [Type MIME JSON](#)
- [Type MIME XML](#)
- [type MIME de l'image](#)
- [type MIME audio ou vidéo](#)
- [police type MIME](#)
- [analyser un type MIME](#)
- [le type MIME est-il pris en charge par l'agent utilisateur ?](#)

## Aller chercher

Les termes suivants sont définis dans *Fetch* : [\[FETCH\]](#)

- [ABNF](#)
- [about:blank](#)
- Un [schéma HTTP\(S\)](#)
- Une URL qui [est locale](#)
- Un [régime local](#)
- Un [schéma de récupération](#)
- [Protocole CORS](#)
- [User-Agent](#) valeur `` par défaut
- [extraire un type MIME](#)
- [héritage extraire un encodage](#)
- [aller chercher](#)
- [récupérer le contrôleur](#)
- [traiter la prochaine redirection manuelle](#)
- [bon état](#)
- [demande de navigation](#)
- [erreur réseau](#)
- [erreur réseau interrompue](#)
- [`Origin`](#) en-tête
- [`Cross-Origin-Resource-Policy`](#) en-tête
- [obtenir une valeur de champ structuré](#)
- [liste d'en-tête](#)



- [ensemble](#)
- [obtenir, décoder et diviser](#)
- [avorter](#)
- [vérification de la stratégie de ressources d'origine croisée](#)
- l' [RequestCredentials](#) énumération
- l' [RequestDestination](#) énumération
- la [fetch\(\)](#) méthode
- [calendrier du rapport](#)
- [sérialiser une URL de réponse pour les rapports](#)
- [extraire un corps en toute sécurité](#)
- [lire progressivement un corps](#)
- [processResponseConsumeBody](#)
- [processResponseEndOfBodyprocessResponseEndOfBody](#)
- [processusRéponse](#)
- [useParallelQueue](#)
- [processEarlyHintsResponse](#)
- [pool de connexion](#)
- [obtenir une connexion](#)
- [déterminer la clé de partition réseau](#)
- [extraire les informations de synchronisation complètes](#)
- [en tant que corps](#)
- [réponse](#) et son associé :
  - [taper](#)
  - [URL](#)
  - [Liste d'URL](#)
  - [statut](#)
  - [liste d'en-tête](#)
  - [corps](#)
  - [informations sur le corps](#)
  - [réponse interne](#)
  - [URL de l'emplacement](#)
  - [infos horaires](#)
  - [informations sur le calendrier des travailleurs de service](#)
  - [a des redirections d'origine croisée](#)
  - [réponse filtrée par redirection opaque](#)
  - [extraire les valeurs de la plage de contenu](#)
- [demande](#) et ses associés :
  - [URL](#)
  - [méthode](#)
  - [liste d'en-tête](#)
  - [corps](#)
  - [client](#)
  - [Liste d'URL](#)
  - [URL actuelle](#)
  - [client réservé](#)
  - [remplace l'identifiant client](#)
  - [initiateur](#)
  - [destination](#)
  - [destination potentielle](#)
  - [traduire](#) une [destination potentielle](#)

- [destinations de type script](#)
- [priorité](#)
- [origine](#)
- [référent](#)
- [drapeau synchrone](#)
- [mode](#)
- [mode d'identification](#)
- [utiliser l'indicateur d'informations d'identification d'URL](#)
- [indicateur de demande non sécurisée](#)
- [mode cache](#)
- [nombre de redirections](#)
- [mode de redirection](#)
- [conteneur de stratégie](#)
- [politique de référence](#)
- [métadonnées nonce cryptographiques](#)
- [métadonnées d'intégrité](#)
- [métadonnées de l'analyseur](#)
- [indicateur de rechargement de navigation](#)
- [indicateur de navigation historique](#)
- [activation de l'utilisateur](#)
- [blocage du rendu](#)
- [type d'initiateur](#)
- [ajouter un en-tête de plage](#)
- [recupérer les informations de synchronisation](#) et ses associés :
  - [Heure de début](#)

Les termes suivants sont définis dans *la politique de parrainage* : [\[REFERRERPOLICY\]](#)

- [politique de référence](#)
- L' [Referrer-Policy](#) en-tête HTTP ``
- L' [analyse d'une politique de référent à partir d'un Referrer-Policy](#) algorithme [d'en-tête ``](#)
- Les stratégies de référence "[no-referrer](#)", "[no-referrer-when-downgrade](#)", "[origin-when-cross-origin](#)" et "[unsafe-url](#)"
- La [politique de référence par défaut](#)

Les termes suivants sont définis dans *le contenu mixte* : [\[MIX\]](#)

- [URL authentifiée a priori](#)

Les termes suivants sont définis dans *Subresource Integrity* : [\[SRI\]](#)

- [analyser les métadonnées d'intégrité](#)
- [obtenir les métadonnées les plus solides de l'ensemble](#)

## Calendrier de peinture

Les termes suivants sont définis dans *Paint Timing* : [\[PAINTTIMING\]](#)

- [marquer le moment de la peinture](#)

## Chronométrage de la navigation

Les termes suivants sont définis dans *Navigation Timing* : [\[NAVIGATIONTIMING\]](#)

- [créer l'entrée de temps de navigation](#)
- [mettre en file d'attente l'entrée de synchronisation de navigation](#)
- [NavigationTimingType](#) et ses valeurs "[navigate](#)", "[reload](#)" et "[back forward](#)"

## Tâches longues

Les termes suivants sont définis dans *les Tâches Longues* : [\[LONGTASKS\]](#)

- [signaler les tâches longues](#)

## IDL Web

Les fragments IDL de cette spécification doivent être interprétés comme requis pour les fragments IDL conformes, comme décrit dans *Web IDL* . [\[WEBIDL\]](#)

Les termes suivants sont définis dans *Web IDL* :

- [ce](#)
- [attribut étendu](#)
- [constructeur nommé](#)
- [opération constructeur](#)
- [étapes du constructeur remplacées](#)
- [créer en interne un nouvel objet implémentant l'interface](#)
- [nom de la propriété d'index de tableau](#)
- [prend en charge les propriétés indexées](#)
- [indices de propriété pris en charge](#)
- [déterminer la valeur d'un bien indexé](#)
- [définir la valeur d'une propriété indexée existante](#)
- [définir la valeur d'une nouvelle propriété indexée](#)
- [prendre en charge les propriétés nommées](#)
- [noms de propriété pris en charge](#)
- [déterminer la valeur d'une propriété nommée](#)
- [définir la valeur d'une propriété nommée existante](#)
- [définir la valeur d'une nouvelle propriété nommée](#)
- [supprimer une propriété nommée existante](#)
- [effectuer un contrôle de sécurité](#)
- [objet plate-forme](#)
- [objet de plate-forme hérité](#)
- [interface principale](#)
- [objet d'interface](#)
- [inclure](#)
- [hériter](#)
- [objet prototype d'interface](#)
- [met en oeuvre](#)
- [Champ \[\[Realm\]\] d'un objet de plateforme](#)
- [contexte de rappel](#)
- [tableau gelé](#) et [création d'un tableau gelé](#)

- [créer un nouvel objet implémentant l'interface](#)
- [rappeler cette valeur](#)
- [conversion](#) entre les types Web IDL et les types JS
- [invoker](#) et [construire](#) des fonctions de rappel
- [algorithme de résolution de surcharge](#)
- [exposé](#)
- [une promesse résolue avec](#)
- [une promesse rejetée avec](#)
- [en cas de rejet](#)
- [à l'accomplissement](#)
- [\[LegacyFactoryFunction\]](#)
- [\[LegacyLenientThis\]](#)
- [\[LegacyNullToEmptyString\]](#)
- [\[LegacyOverrideBuiltIns\]](#)
- [\[LegacyTreatNonObjectAsNull\]](#)
- [\[LegacyUnenumerableNamedProperties\]](#)
- [\[LegacyUnforgeable\]](#)

Web IDL définit également les types suivants qui sont utilisés dans les fragments Web IDL dans cette spécification :

- [ArrayBuffer](#)
- [ArrayBufferView](#)
- [boolean](#)
- [DOMString](#)
- [double](#)
- [énumération](#)
- [Function](#)
- [long](#)
- [object](#)
- [Uint8ClampedArray](#)
- [unrestricted double](#)
- [unsigned long](#)
- [USVString](#)
- [VoidFunction](#)

Le terme [lancer](#) dans cette spécification est utilisé tel que défini dans *Web IDL* . Le [DOMException](#) type et les noms d'exception suivants sont définis par Web IDL et utilisés par cette spécification :

- ["IndexSizeError"](#)
- ["HierarchyRequestError"](#)
- ["InvalidCharacterError"](#)
- ["NoModificationAllowedError"](#)
- ["NotFoundError"](#)
- ["NotSupportedError"](#)
- ["InvalidStateError"](#)
- ["SyntaxError"](#)
- ["InvalidAccessError"](#)
- ["SecurityError"](#)
- ["NetworkError"](#)
- ["AbortError"](#)
- ["QuotaExceededError"](#)

- ["DataCloneError"](#)
- ["EncodingError"](#)
- ["NotAllowedError"](#)

Lorsque cette spécification exige qu'un agent utilisateur **crée un `Date` objet** représentant un temps particulier (qui pourrait être la valeur spéciale Not-a-Number), le composant millisecondes de ce temps, le cas échéant, doit être tronqué à un entier, et la valeur de temps de l' `Date` objet nouvellement créé doit représenter l'heure tronquée résultante.

Par exemple, étant donné l'heure 23045 millièmes de seconde après 01:00 UTC le 1er janvier 2000, soit l'heure 2000-01-01T00:00:00.023045Z, alors l'objet créé `Date` représentant cette heure représenterait la même heure que celle créée représentant l'heure 2000-01-01T00:00:00.023Z, 45 millièmes plus tôt. Si le temps donné est NaN, alors le résultat est un `Date` objet qui représente une valeur de temps NaN (indiquant que l'objet ne représente pas un instant de temps spécifique).

## JavaScript

Certaines parties du langage décrit par cette spécification ne prennent en charge que JavaScript comme langage de script sous-jacent. [\[JAVASCRIPT\]](#)

*Le terme "JavaScript" est utilisé pour désigner ECMA-262, plutôt que le terme officiel ECMAScript, puisque le terme JavaScript est plus largement connu.*

Les termes suivants sont définis dans la spécification JavaScript et utilisés dans cette spécification :

- [objet de fonction actif](#)
- [agent](#) et [groupe d'agents](#)
- [insertion automatique de points-virgules](#)
- [exécution du candidat](#)
- Le [royaume actuel](#)
- [erreur précoce](#)
- [progrès vers l'avant](#)
- [invariants des méthodes internes essentielles](#)
- [Contexte d'exécution JavaScript](#)
- [Pile de contexte d'exécution JavaScript](#)
- [royaume](#)
- [Enregistrement de rappel de travail](#)
- [Nouvelle cible](#)
- [exécution du contexte d'exécution JavaScript](#)
- [agent environnant](#)
- [fermeture abstraite](#)
- [objet exotique prototype immuable](#)
- [Symboles bien connus](#) , y compris `@@hasInstance` , `@@isConcatSpreadable` , `@@toPrimitive` et `@@toStringTag`
- [Objets intrinsèques bien connus](#) , y compris `%Array.prototype%` , `%Error.prototype%` , `%EvalError.prototype%` , `%Function.prototype%` , `%JSON.parse%` , `%Object.prototype%` , `%Object.prototype.valueOf%` , `%RangeError.prototype%` ,

**%ReferenceError.prototype%** , **%SyntaxError.prototype%** , **%TypeError.prototype%** et **%URIError.prototype%**

- La production [FunctionBody](#)
- La fabrication [de modules](#)
- La fabrication [du patron](#)
- La production [du scénario](#)
- La notation [Type](#)
- Le type de spécification [d'enregistrement d'achèvement](#)
- Les types de spécification [List](#) et [Record](#)
- Le type de spécification [de descripteur de propriété](#)
- Le type de spécification [d'enregistrement de script](#)
- Le type de spécification [d'enregistrement de module cyclique](#)
- Le type de spécification [Source Text Module Record](#) et ses méthodes [Evaluate](#) , [Link](#) et [LoadRequestedModules](#)
- L' opération abstraite [ArrayCreate](#)
- L' opération abstraite [d'appel](#)
- L' opération abstraite [ClearKeptObjects](#)
- L' opération abstraite [CleanupFinalizationRegistry](#)
- L' opération abstraite [Construire](#)
- L' opération abstraite [CopyDataBlockBytes](#)
- L' opération abstraite [CreateBuiltinFunction](#)
- L' opération abstraite [CreateByteDataBlock](#)
- L' opération abstraite [CreateDataProperty](#)
- L' opération abstraite [DetachArrayBuffer](#)
- L' opération abstraite [EnumerableOwnProperties](#)
- L' opération abstraite [FinishDynamicImport](#)
- L' opération abstraite [FinishLoadingImportedModule](#)
- L' opération abstraite [OrdinaryFunctionCreate](#)
- L' opération [Get](#) abstract
- L' opération abstraite [GetActiveScriptOrModule](#)
- L' opération abstraite [GetFunctionRealm](#)
- L' opération abstraite [HasOwnProperty](#)
- L' opération abstraite [HostCallJobCallback](#)
- Opération [abstraite](#) [HostEnqueueFinalizationRegistryCleanupJob](#)
- L' opération abstraite [HostEnqueuePromiseJob](#)
- L' opération abstraite [HostEnsureCanAddPrivateElement](#)
- L' opération abstraite [HostEnsureCanCompileStrings](#)
- L' opération abstraite [HostLoadImportedModule](#)
- L' opération abstraite [HostMakeJobCallback](#)
- L' opération abstraite [HostPromiseRejectionTracker](#)
- L' opération abstraite [InitializeHostDefinedRealm](#)
- L' opération abstraite [IsAccessorDescriptor](#)
- L' opération abstraite [IsCallable](#)
- L' opération abstraite [IsConstructor](#)
- L' opération abstraite [IsDataDescriptor](#)
- L' opération abstraite [IsDetachedBuffer](#)
- L' opération abstraite [IsSharedArrayBuffer](#)
- L' opération abstraite [NewObjectEnvironment](#)
- L' opération abstraite [NormalCompletion](#)
- L' opération abstraite [OrdinaryGetPrototypeOf](#)

- L'opération abstraite [OrdinarySetPrototypeOf](#)
- L'opération abstraite [OrdinaryIsExtensible](#)
- L'opération abstraite [OrdinaryPreventExtensions](#)
- L'opération abstraite [OrdinaryGetOwnProperty](#)
- L'opération abstraite [OrdinaryDefineOwnProperty](#)
- L'opération abstraite [OrdinaryGet](#)
- L'opération abstraite [OrdinarySet](#)
- L'opération abstraite [OrdinaryDelete](#)
- L'opération abstraite [OrdinaryOwnPropertyKeys](#)
- L'opération abstraite [OrdinaryObjectCreate](#)
- L'opération abstraite [ParseModule](#)
- L'opération abstraite [ParseScript](#)
- L'opération abstraite [NewPromiseReactionJob](#)
- L'opération abstraite [NewPromiseResolveThenableJob](#)
- L'opération abstraite [RegExpBuiltinExec](#)
- L'opération abstraite [RegExpCreate](#)
- L'opération abstraite [RunJobs](#)
- L'opération abstraite [SameValue](#)
- L'opération abstraite [ScriptEvaluation](#)
- L'opération abstraite [SetImmutablePrototype](#)
- L'opération abstraite [ToBoolean](#)
- L'opération abstraite [ToString](#)
- L'opération abstraite [ToUint32](#)
- L'opération abstraite [TypedArrayCreate](#)
- L'opération abstraite [IsLooselyEqual](#)
- L'opération abstraite [IsStrictlyEqual](#)
- L' [Atomics](#) objet
- La [Date](#) classe
- La [FinalizationRegistry](#) classe
- La [RegExp](#) classe
- La [SharedArrayBuffer](#) classe
- La [TypeError](#) classe
- La [RangeError](#) classe
- La [WeakRef](#) classe
- La [eval\(\)](#) fonction
- La [WeakRef.prototype.deref\(\)](#) fonction
- L'emplacement interne [\[IsHTMLDDA\]](#)
- [import\(\)](#)
- [import.meta](#)
- L'opération abstraite [HostGetImportMetaProperties](#)
- L' [typeof](#) opérateur
- L' [delete](#) opérateur
- Le tableau [des constructeurs TypedArray](#)

Les agents utilisateurs prenant en charge JavaScript doivent également implémenter l'API d'internationalisation ECMAScript . [\[JSINTL\]](#)

Les agents utilisateurs qui prennent en charge JavaScript doivent également implémenter la proposition d'importation d'assertions . Les termes suivants y sont définis et utilisés dans cette spécification : [\[JSIMPORTASSERTIONS\]](#)



- Le type de spécification [ModuleRequest Record](#)
- L'opération abstraite [HostGetSupportedImportAssertions](#)

Les agents utilisateurs prenant en charge JavaScript doivent également implémenter la proposition de *modules JSON*. Les termes suivants y sont définis et utilisés dans cette spécification : [\[JSJSONMODULES\]](#)

- L'opération abstraite [CreateDefaultExportSyntheticModule](#)
- L'opération abstraite [SetSyntheticModuleExport](#)
- Le type de spécification [d'enregistrement de module synthétique](#)
- L'opération abstraite [ParseJSONModule](#)

Les agents utilisateurs qui prennent en charge JavaScript doivent également implémenter les propositions *Resizable ArrayBuffer* et *Growable SharedArrayBuffer*. Les termes suivants y sont définis et utilisés dans cette spécification : [\[JSRESIZABLEBUFFERS\]](#)

- L'opération abstraite [IsArrayBufferViewOutOfBounds](#)

## WebAssembly

Le terme suivant est défini dans *WebAssembly JavaScript Interface* : [\[WASMJS\]](#)

- [WebAssembly.Module](#)

## DOM

Le Document Object Model (DOM) est une représentation — un modèle — d'un document et de son contenu. Le DOM n'est pas qu'une API ; les critères de conformité des implémentations HTML sont définis, dans cette spécification, en termes d'opérations sur le DOM. [\[DOM\]](#)

Les implémentations doivent prendre en charge DOM et les événements définis dans les événements d'interface utilisateur, car cette spécification est définie en termes de DOM, et certaines des fonctionnalités sont définies comme des extensions des interfaces DOM. [\[DOM\]](#) [\[UIEVENTS\]](#)

En particulier, les fonctionnalités suivantes sont définies dans *DOM* : [\[DOM\]](#)

- [Attr](#)interface
- [CharacterData](#)interface
- [Comment](#)interface
- [DOMImplementation](#)interface
- [Document](#)interface et son [doctype](#)attribut
- [DocumentOrShadowRoot](#)interface
- [DocumentFragment](#)interface
- [DocumentType](#)interface
- [ChildNode](#)interface
- [Element](#)interface
- [attachShadow\(\)](#) méthode.
- [La racine fantôme](#) d'un élément
- L' [algorithme de recilage](#)
- [Node](#)interface



- NodeListinterface
- ProcessingInstructioninterface
- ShadowRootinterface
- Textinterface
- concept de document de nœud
- notion de type de document
- notion d'hôte
- Le concept de racine fantôme et ses délégés se concentrent et sont disponibles pour les éléments internes .
- Le concept d'hôte fantôme
- HTMLCollectioninterface, son lengthattribut et item() ses namedItem() méthodes
- Les termes collection et représenté par la collection
- DOMTokenListinterface, et son valueattribut et supportsopération
- createDocument() méthode
- createHTMLDocument() méthode
- createElement() méthode
- createElementNS() méthode
- getElementById() méthode
- getElementsByClassName() méthode
- appendChild() méthode
- cloneNode() méthode
- importNode() méthode
- preventDefault() méthode
- idattribut
- setAttribute() méthode
- textContentattribut
- Les concepts d'arbre , d'arbre fantôme et d'arbre de nœuds
- Les concepts d'ordre des arbres et d'ordre des arbres avec ombre
- La notion d'élément
- La notion d'enfant
- Les concepts de racine et d'ombre incluant la racine
- Les concepts d'ancêtre inclusif , de descendant , d'ancêtre inclus dans l'ombre , de descendant inclus dans l'ombre , de descendant inclus inclus dans l'ombre et d'ancêtre inclus inclus dans l'ombre
- Le premier enfant , le frère suivant et les concepts de frère précédent
- Le concept d'élément parent
- Le concept d'élément de document
- Dans une arborescence de documents , dans un document (hérité) et concepts connexes
- Le concept de slot , et son nom et les nœuds assignés
- Le concept d'emplacement attribué
- Le concept d'affectation des créneaux
- Le concept d'encoche
- L' assignation des slottables pour un algorithme d'arborescence
- L' slotchangeévénement
- Le concept de descendant inclusif
- L' algorithme de recherche de slottables aplaties
- Le concept d'affectation manuelle des créneaux

- L' algorithme d'attribution d'un emplacement
- Les algorithmes pre-insert , insert , append , replace , replace all , string replace all , remove et adopt pour les nœuds
- La notion de descendance
- Les étapes d'insertion , de suppression d'étapes , d'adoption d'étapes et d'étapes enfants ont modifié les crochets des éléments.
- Les algorithmes de modification , d'ajout , de suppression , de remplacement et de définition de valeur pour les attributs
- Le crochet des étapes de changement d'attribut pour les attributs
- Le concept de liste d'attributs
- Les données d'un `CharacterData` nœud et son algorithme de remplacement des données
- Le contenu du texte enfant d'un nœud
- Le contenu textuel descendant d'un nœud
- Le nom , l'ID public et l'ID système d'un doctype
- Event interface
- Event et comportement du constructeur d'interfaces dérivées
- EventTarget interface
- Le crochet de comportement d'activation
- Le hook de comportement hérité de pré-activation
- Le hook de comportement legacy-canceled-activation
- L' algorithme de création d'un événement
- L' algorithme d'incendie et d'événement
- Le drapeau annulé
- L' algorithme de répartition
- EventInit Sorte de dictionnaire
- type attribut
- La cible d'un événement
- currentTarget attribut
- bubbles attribut
- cancelable attribut
- composed attribut
- drapeau composé
- isTrusted attribut
- initEvent() méthode
- ajouter un écouteur d'événement
- addEventListener() méthode
- Les algorithmes de suppression d'un écouteur d'événement et de suppression de tous les écouteurs d'événement
- EventListener interface de rappel
- Le type d'événement
- Un écouteur d'événement , son type et son rappel
- L' encodage (ici l' *encodage de caractères* ), le mode et le type de contenu d'un `Document`
- La distinction entre les documents XML et les documents HTML
- Les termes mode bizarrerie , mode bizarrerie limitée et mode sans bizarrerie
- L'algorithme pour cloner un `Node`, et le concept d' étapes de clonage utilisé par cet algorithme

- Le concept des **étapes de changement d'URL de base** et la définition de ce qui se passe lorsqu'un élément est **affecté par un changement d'URL de base**
- Le concept d' identifiant unique (ID) d'un élément
- Le concept des classes d'un élément
- Le terme jetons pris en charge
- Le concept d'une plage DOM et les termes start , end et border point appliqués aux plages.
- L' algorithme de création d'un élément
- Le concept d'interface d'élément
- Les concepts d' état d'élément personnalisé et d' éléments définis et personnalisés
- Espace de noms , préfixe d'espace de noms , nom local , définition d'élément personnalisé et isvaleur d'un élément
- MutationObserver interface et observateurs de mutations en général
- L' algorithme d' obtention d'un attribut par nom

Les fonctionnalités suivantes sont définies dans *les événements de l'interface utilisateur* : [UIEVENTS]

- L' MouseEvent interface
- L' attribut MouseEvent de l'interface relatedTarget
- MouseEventInit Sorte de dictionnaire
- L' FocusEvent interface
- L' attribut FocusEvent de l'interface relatedTarget
- L' UIEvent interface
- L' attribut UIEvent de l'interface view
- auxclick événement
- beforeinput événement
- click événement
- contextmenu événement
- dblclick événement
- input événement
- mousedown événement
- mouseenter événement
- mouseleave événement
- mousemove événement
- mouseout événement
- mouseover événement
- mouseup événement
- wheel événement
- keydown événement
- keypress événement
- keyup événement

Les fonctionnalités suivantes sont définies dans *les événements tactiles* : [TOUCH]

- Touch interface
- Concept de point de contact
- touchend événement

Les fonctionnalités suivantes sont définies dans *Pointer Events* : [\[POINTEREVENTS\]](#)

- L' [PointerEvent](#) interface
- L' attribut [PointerEvent](#) de l'interface [pointerType](#)
- [déclencher un événement de pointeur](#)
- [pointerdown](#) événement
- [pointerup](#) événement
- [pointercancel](#) événement

Les événements suivants sont définis dans *l'API et les événements du Presse-papiers* : [\[CLIPBOARD-APIS\]](#)

- [copy](#) événement
- [cut](#) événement
- [paste](#) événement

Cette spécification utilise parfois le terme **name** pour désigner le [type](#) d'événement ; comme dans "un événement nommé `click`" ou "si le nom de l'événement est `keypress`". Les termes « nom » et « type » pour les événements sont synonymes.

Les fonctionnalités suivantes sont définies dans *DOM Parsing and Serialization* : [\[DOMPARSING\]](#)

- [innerHTML](#)
- [outerHTML](#)

Les fonctionnalités suivantes sont définies dans *l'API de sélection* : [\[SELECTION\]](#)

- [sélection](#)
- [Selection](#)

*Les agents utilisateurs sont encouragés à implémenter les fonctionnalités décrites dans `execCommand` .* [\[EXECCOMMANDE\]](#)

Les parties suivantes de *l'API Fullscreen* sont référencées à partir de cette spécification, en partie pour définir le rendu des `dialog` éléments, et également pour définir comment l'API Fullscreen interagit avec HTML : [\[FULLSCREEN\]](#)

- [couche supérieure](#) (un [ensemble ordonné](#) ) et son opération [d'ajout](#)
- [requestFullscreen\(\)](#)
- [exécuter les étapes plein écran](#)
- [drapeau plein écran](#)

*High Resolution Time* fournit l' [heure haute résolution actuelle](#) , l' [heure actuelle partagée non sécurisée](#) , l' [horloge monotone partagée](#) , l' algorithme [de temps grossier](#) [DOMHighResTimeStamp](#) et le typedef. [\[THS\]](#)

## API de fichier

Cette spécification utilise les fonctionnalités suivantes définies dans *File API* : [\[FILEAPI\]](#)

- L' [Blob](#) interface et son [type](#) attribut

- L' [File](#) interface et [name](#) ses [lastModified](#) attributs
- L' [FileList](#) interface
- Le concept d' [état d'instantané](#) [Blob](#) de
- Le concept d' [erreurs de lecture](#)
- [Magasin d'URL Blob](#)
- [entrée d'URL blob](#) et son [objet](#) et [son environnement](#)

## API de base de données indexée

Cette spécification utilise [les transactions de nettoyage de base de données indexées](#) définies par l'API de base de données indexée . [\[INDEXEDDB\]](#)

## Extensions de source multimédia

Les termes suivants sont définis dans *les extensions de source multimédia* : [\[MEDIASOURCE\]](#)

- [MediaSource](#) interface
- [se détacher d'un élément multimédia](#)

## Capture multimédia et flux

Les termes suivants sont définis dans *Media Capture and Streams* : [\[MEDIASTREAM\]](#)

- [MediaStream](#) interface

## Rapports

Les termes suivants sont définis dans *Reporting* : [\[REPORTING\]](#)

- [Mettre un rapport en file d'attente](#)
- [type de rapport](#)
- [visible pour](#) [ReportingObservers](#)

## XMLHttpRequest

Les fonctionnalités et termes suivants sont définis dans *XMLHttpRequest* : [\[XHR\]](#)

- L' [XMLHttpRequest](#) interface et son [responseXML](#) attribut
- L' [ProgressEvent](#) interface et ses attributs [lengthComputable](#), [loaded](#) et [total](#)
- L' [FormData](#) interface et sa [liste d'entrées associée](#)

## État de la batterie

Les fonctionnalités suivantes sont définies dans l'API d'état de la batterie : [\[BATTERIE\]](#)

- [getBattery\(\)](#) méthode

## Requêtes multimédias

Les implémentations doivent prendre en charge *les Media Queries* . La caractéristique [<media-condition>](#) y est définie. [\[QM\]](#)

## Modules CSS

Bien que la prise en charge de CSS dans son ensemble ne soit pas requise pour les implémentations de cette spécification (bien qu'elle soit encouragée, du moins pour les navigateurs Web), certaines fonctionnalités sont définies en termes d'exigences CSS spécifiques.

Lorsque cette spécification exige que quelque chose soit [analysé selon une grammaire CSS particulière](#), l'algorithme pertinent dans *la syntaxe CSS* doit être suivi, y compris les règles de gestion des erreurs. [\[CSSSYNTAXE\]](#)

Par exemple, les agents utilisateurs doivent fermer toutes les constructions ouvertes lorsqu'ils trouvent la fin d'une feuille de style de manière inattendue. Ainsi, lors de l'analyse de la chaîne " `rgb(0,0,0` " (avec une parenthèse fermante manquante) pour une valeur de couleur, la parenthèse fermante est impliquée par cette règle de gestion des erreurs, et une valeur est obtenue (la couleur 'noir'). Cependant, la construction similaire " `rgb(0,0,` " (avec à la fois une parenthèse manquante et une valeur "bleue" manquante) ne peut pas être analysée, car la fermeture de la construction ouverte ne donne pas une valeur viable.

Pour **analyser une valeur CSS <color>**, étant donné une *entrée* de chaîne avec un élément facultatif *élément* optionnel *element*, exécutez ces étapes :

1. Soit *color* le résultat de [l'analyse](#) de *l'entrée* en tant que CSS [<color>](#). [\[CSSCOLOR\]](#)
2. Si *la couleur* est un échec, renvoie l'échec.
3. Si *la couleur* est ['currentcolor'](#), alors :
  1. Si *l'élément* n'est pas donné, alors définissez *la couleur* sur [noir opaque](#).
  2. Sinon, définissez *color* sur la valeur calculée de la propriété ['color'](#) de *l'élément*.
4. Retour *Couleur*.

Les termes et fonctionnalités suivants sont définis dans *les feuilles de style en cascade* ( CSS ) : [\[CSS\]](#)

- [fenêtre](#)
- [boîte de ligne](#)
- [hors flux](#)
- [afflux](#)
- [effondrement des marges](#)
- [bloc contenant](#)
- [boîte en ligne](#)
- [boîte de bloc](#)
- Le ['haut'](#), ['bas'](#), ['gauche'](#) et propriétés ['right'](#)
- Le propriété ["flottante"](#)

- Le propriété ["clair"](#)
- Le propriété ["largeur"](#)
- Le propriété ["hauteur"](#)
- Le propriété ["max-width"](#)
- Le propriété ["max-height"](#)
- La ["hauteur de ligne"](#) propriété
- Le ["vertical-align"](#) propriété
- Le ["contenu"](#) propriété
- La valeur ["inline-block"](#) de la propriété ["display"](#)
- La propriété ["visibilité"](#)

La version de base de la propriété ["display"](#) est définie dans CSS , et la propriété est étendue par d'autres modules CSS. [\[CSS\]](#) [\[CSSRUBY\]](#) [\[CSSTABLE\]](#)

Les termes et fonctionnalités suivants sont définis dans CSS Box Model : [\[CSSBOX\]](#)

- [zone de contenu](#)
- [zone de contenu](#)
- [boîte de bordure](#)
- [boîte de marge](#)
- [bord de la frontière](#)
- [bord de la marge](#)
- ["margin-top"](#) , ["margin-bottom"](#) , ["margin-left"](#) et ["margin-right"](#) propriétés
- Les propriétés ["padding-top"](#) , ["padding-bottom"](#) , ["padding-left"](#) et ["padding-right"](#)

Les fonctionnalités suivantes sont définies dans *les propriétés logiques* CSS : [\[CSSLOGICAL\]](#)

- ' margin [-block-start"](#) , ["margin-block-end"](#) , ["margin-inline-start"](#) , et propriétés ["margin-inline-end"](#)
- ' padding [-block-start"](#) , ["padding-block-end"](#) , ["padding-inline-start"](#) et ["padding-inline-end"](#) propriétés
- Le ["border-block-start-width"](#) , ["border-block-end-width"](#) , ["border-inline-start-width"](#) , ["border-inline-end-width"](#) , ["border-block-start-style"](#) , ["border-block-end-style"](#) , ["border-inline-start-style"](#) , ["border-inline-end-style"](#) , ["border-block-start-color"](#) , ["border-block-end-color"](#) , ["couleur de début de bordure en ligne"](#) , ["couleur de fin de bordure en ligne"](#) , ["frontière-début-début-rayon"](#) Propriétés , ["border-start-end-radius"](#) , ["border-end-start-radius"](#) et ["border-end-end-radius"](#)
- La propriété ["taille de bloc"](#)

- La propriété ['inline-size'](#)
- La propriété ['inset-block-start'](#)
- La propriété ['inset-block-end'](#)

Les termes et fonctionnalités suivants sont définis dans CSS

Color : [\[CSSCOLOR\]](#)

- [couleur nommée](#)
- [<couleur>](#)
- La propriété ['couleur'](#)
- La valeur ['currentcolor'](#)
- [noir opaque](#)
- [noir transparent](#)
- espace colorimétrique ['srgb'](#)
- espace colorimétrique ['display-p3'](#)
- Intention de rendu ["colorimétrique relatif"](#)

Les termes suivants sont définis dans CSS Images : [\[CSSIMAGES\]](#)

- [taille d'objet par défaut](#)
- [dimensions intrinsèques](#)
- [hauteur intrinsèque](#)
- [largeur intrinsèque](#)
- La propriété ['orientation de l'image'](#)
- ['dégradé conique'](#)
- La propriété ['object-fit'](#)

Le terme [source de peinture](#) est utilisé tel que défini dans *CSS Images Level 4* pour définir l'interaction de certains éléments HTML avec la fonction CSS 'element()'. [\[CSSIMAGES4\]](#)

Les fonctionnalités suivantes sont définies dans *CSS Backgrounds and Borders* : [\[CSSBG\]](#)

- Les propriétés ['background-color'](#) , ['background-image'](#) , ['background-repeat'](#) , ['background-attachment'](#) , ['background-position'](#) , ['background-clip'](#) , ['background-origin'](#) et ['background-size'](#)
- Le ['border-radius'](#) , ['border-top-left-radius'](#) , ['border-top-right-radius'](#) , ['border-bottom-right-radius'](#) , ['border-bottom-left-radius'](#) propriétés
- Les propriétés ['border-image-source'](#) , ['border-image-slice'](#) , ['border-image-width'](#) , ['border-image-outset'](#) et ['border-image-repeat'](#)

*CSS Backgrounds and Borders* définit également les propriétés de bordure suivantes : [\[CSSBG\]](#)

Propriétés de bordure

	Haut	Bas	Gauche	Droite
Largeur	<a href="#">'border-top-width'</a>	<a href="#">'border-bottom-width'</a>	<a href="#">'largeur-bordure-gauche'</a>	<a href="#">'border-right-width'</a>



Style	<a href="#"><u>'border-top-style'</u></a>	<a href="#"><u>'style bordure inférieure'</u></a>	<a href="#"><u>'border-left-style'</u></a>	<a href="#"><u>'border-right-style'</u></a>
Couleur	<a href="#"><u>'border-top-color'</u></a>	<a href="#"><u>'border-bottom-color'</u></a>	<a href="#"><u>'border-left-color'</u></a>	<a href="#"><u>'border-right-color'</u></a>

Les fonctionnalités suivantes sont définies dans *CSS Box*

*Alignment* : [\[CSSALIGN\]](#)

- La propriété ['align-content'](#)
- La propriété ['align-items'](#)
- La propriété ['align-self'](#)
- La propriété ["se justifier"](#)
- Le ['justifier-contenu'](#) propriété
- La propriété ['justify-items'](#)

Les termes et fonctionnalités suivants sont définis dans *CSS*

*Display* : [\[CSSDISPLAY\]](#)

- La propriété ['align-content'](#)
- La propriété ['align-items'](#)
- La propriété ['align-self'](#)
- La propriété ["se justifier"](#)
- La propriété ['justify-content'](#)
- La propriété ['justify-items'](#)

Les termes et fonctionnalités suivants sont définis dans *CSS*

*Display* : [\[CSSDISPLAY\]](#)

- [type d'affichage externe](#)
- [type d'affichage intérieur](#)
- [au niveau du bloc](#)
- [bloc conteneur](#)
- [contexte de formatage](#)
- [contexte de formatage de bloc](#)
- [contexte de formatage en ligne](#)
- [élément remplacé](#)
- [Boîte CSS](#)

Les fonctionnalités suivantes sont définies dans *CSS Flexible Box*

*Layout* : [\[CSSFLEXBOX\]](#)

- La propriété ['flex-direction'](#)
- La propriété ['flex-wrap'](#)

Les termes et fonctionnalités suivants sont définis dans *CSS*

*Fonts* : [\[CSSFONTS\]](#)

- [première police disponible](#)
- La ["famille de polices"](#) propriété
- Le ['poids de la police'](#) propriété
- La ["taille de la police"](#) propriété
- La ["police"](#) propriété
- Le ["crénage de police"](#) propriété

- La "font-stretch" propriété
- Les 'font-variant-caps' propriété
- Les "petites capitalisations" valeur
- Le "tout petites capitalisations" valeur
- Les "petites casquettes" valeur
- La valeur "tout en petites majuscules"
- La valeur 'unicase'
- La valeur 'titling-caps'
- Le "ultra-condensé" valeur
- Le 'extra-condensé' valeur
- Le "condensé" valeur
- Le "semi-condensé" valeur
- Le "semi-expansé" valeur
- Le "développé" valeur
- Le "extra-élargi" valeur
- La valeur "ultra-élargie"

Les fonctionnalités suivantes sont définies dans *CSS Grid Layout* : [\[CSSGRID\]](#)

- La 'grille-auto-colonnes' propriété
- Le 'grid-auto-flow' propriété
- La 'grille-auto-lignes' propriété
- La propriété 'grid-column-gap'
- La propriété 'grid-row-gap'
- La propriété 'grid-template-areas'
- La propriété 'grid-template-columns'
- La propriété 'grid-template-rows'

Les termes suivants sont définis dans *CSS Inline Layout* : [\[CSSINLINE\]](#)

- ligne de base alphabétique
- métrique d'ascension
- métrique de descente
- ligne de base suspendue
- idéographique sous ligne de base

Les termes et fonctionnalités suivants sont définis dans *CSS Intrinsic & Extrinsic Sizing* : [\[CSSSIZING\]](#)

- taille en ligne fit-content
- propriété 'rapport d'aspect'

Les fonctionnalités suivantes sont définies dans *CSS Lists and Counters* . [\[LISTES CSS\]](#)

- élément de liste
- La propriété 'counter-reset'
- Le "contre-ensemble" propriété
- La propriété 'list-style-type'

Les fonctionnalités suivantes sont définies dans *CSS Overflow* . [\[CSSOVERFLOW\]](#)

- La propriété 'overflow' et son 'caché' valeur

- Le ["débordement de texte"](#) propriété
- Le terme [conteneur de défilement](#)

Les termes et fonctionnalités suivants sont définis dans *CSS Positioned Layout* : [\[CSSPOSITION\]](#)

- [absolument positionné](#)
- La propriété ['position'](#) et sa valeur ['statique'](#)

Les fonctionnalités suivantes sont définies dans *CSS Multi-column Layout* . [\[CSSMULTICOL\]](#)

- Le ["compte de colonnes"](#) propriété
- Le ["remplissage de colonne"](#) propriété
- Le [« trou de colonne »](#) propriété
- La ["règle de la colonne"](#) propriété
- La propriété ["largeur de colonne"](#)

La valeur ['ruby-base'](#) de la propriété ['display'](#) est définie dans *CSS Ruby Layout* . [\[CSSRUBY\]](#)

Les fonctionnalités suivantes sont définies dans *CSS Table* : [\[CSSTABLE\]](#)

- La propriété ['border-spacing'](#)
- La propriété ['border-collapse'](#)
- Les valeurs ['table-cell'](#) , ['table-row'](#) , ['table-caption'](#) et ['table'](#) de la propriété ['display'](#)

Les fonctionnalités suivantes sont définies dans *CSS Text* : [\[CSSTEXT\]](#)

- La propriété ['text-transform'](#)
- La propriété ['espace blanc'](#)
- La propriété ['text-align'](#)
- L' ["espacement des lettres"](#) propriété
- La propriété ['word-spacing'](#)

Les fonctionnalités suivantes sont définies dans *les modes d'écriture CSS* : [\[CSSWM\]](#)

- La propriété ['direction'](#)
- La propriété ['unicode-bidi'](#)
- Les concepts [de sens de flux de bloc](#) , [d'axe de bloc](#) , [d'axe en ligne](#) , [de taille de bloc](#) , [de taille en ligne](#) , [de début de bloc](#) , [de fin de bloc](#) , [de début en ligne](#) , de [fin en ligne](#) , [de ligne gauche](#) et [de ligne droite](#)

Les fonctionnalités suivantes sont définies dans *CSS Basic User Interface* : [\[CSSUI\]](#)

- La propriété ['contour'](#)
- La propriété ['curseur'](#)
- La propriété ['appearance'](#) , son type de valeur non terminale [<compat-auto>](#) , sa valeur ['textfield'](#) et sa valeur ['menulist-button'](#) .
- [Le widget](#) conceptuel
- Le concept [d'apparence native](#)
- Le concept [d'apparence primitive](#)

- La classification [des widgets non dévolus](#) et [des widgets dévolus](#) , et l' état [des widgets dévolus associés](#).
- La propriété ['pointer-events'](#)
- La propriété ['user-select'](#)

L'algorithme de [mise à jour des animations et d'envoi d'événements](#) est défini dans *Web Animations* . [\[WEBANIMATIONS\]](#) .

Les implémentations qui prennent en charge les scripts doivent prendre en charge le modèle d'objet CSS. Les fonctionnalités et termes suivants sont définis dans les spécifications CSSOM : [\[CSSOM\]](#) [\[CSSOMVIEW\]](#)

- [Screen](#)interface
- [LinkStyle](#)interface
- [CSSStyleDeclaration](#)interface
- [style](#)Attribut IDL
- [cssText](#)attribut de[CSSStyleDeclaration](#)
- [StyleSheet](#)interface
- [CSSStyleSheet](#)interface
- [créer une feuille de style CSS](#)
- [supprimer une feuille de style CSS](#)
- [feuille de style CSS associée](#)
- [créer un construit](#)[CSSStyleSheet](#)
- [remplacer de manière synchrone les règles d'un](#)[CSSStyleSheet](#)
- [Feuilles de style CSS](#) et leurs propriétés :
  - [taper](#)
  - [emplacement](#)
  - [feuille de style CSS mère](#)
  - [nœud propriétaire](#)
  - [règle CSS du propriétaire](#)
  - [médias](#)
  - [titre](#)
  - [drapeau alternatif](#)
  - [drapeau désactivé](#)
  - [Règles CSS](#)
  - [drapeau d'origine propre](#)
- [Jeu de feuilles de style CSS](#)
- [Nom du jeu de feuilles de style CSS](#)
- [nom du jeu de feuilles de style CSS préféré](#)
- [modifier le nom du jeu de feuilles de style CSS préféré](#)
- [Sérialisation d'une valeur CSS](#)
- [exécuter les étapes de redimensionnement](#)
- [exécuter les étapes de défilement](#)
- [évaluer les requêtes des médias et signaler les modifications](#)
- [Faire défiler un élément dans la vue](#)
- [Faites défiler jusqu'au début du document](#)
- L' [resize](#)événement
- L' [scroll](#)événement
- L' [scrollend](#)événement
- [configurer les fonctionnalités de contexte de navigation](#)

Les fonctionnalités et termes suivants sont définis dans *la syntaxe*  
CSS : [\[CSSSYNTAX\]](#)

- [feuille de style conforme](#)
- [analyser une liste de valeurs de composants](#)
- [analyser une liste de valeurs de composants séparées par des virgules](#)
- [valeur du composant](#)
- [encodage d'environnement](#)
- [<jeton d'espace blanc>](#)

Les termes suivants sont définis dans *les sélecteurs* : [\[SELECTORS\]](#)

- [sélecteur de type](#)
- [sélecteur d'attribut](#)
- [pseudo-classe](#)
- [:focus-visible](#) pseudo-classe
- [indiquer la mise au point](#)
- [pseudo-élément](#)

Les fonctionnalités suivantes sont définies dans *les valeurs et unités*  
CSS : [\[CSSVALUES\]](#)

- [<longueur>](#)
- L'unité ['em'](#)
- L'unité ["ex"](#)
- L'unité ['vw'](#)
- L'unité ["in"](#)
- L'unité ['px'](#)
- L'unité ['pt'](#)
- La fonction ['attr\(\)'](#)
- Les [fonctions mathématiques](#)

Le terme [attribut de style](#) est défini dans *CSS Style Attributes* . [\[CSSATTR\]](#)

Les termes suivants sont définis dans le *CSS Cascading and Héritage* : [\[CSSCASCADE\]](#)

- [valeur en cascade](#)
- [valeur spécifiée](#)
- [valeur calculée](#)
- [valeur d'usage](#)
- [origine cascade](#)
- [Origine de l'auteur](#)
- [Origine de l'utilisateur](#)
- [Origine de l'agent utilisateur](#)
- [Origine des animations](#)
- [Origine de la transition](#)
- [valeur initiale](#)

L' [CanvasRenderingContext2D](#) utilisation des polices par l'objet dépend des fonctionnalités décrites dans les spécifications CSS *Fonts* et *Font Loading* , incluant notamment **FontFace** les objets et le concept [de source de police](#) . [\[CSSFONTS\]](#) [\[CSSFONTLOAD\]](#)

Les interfaces et termes suivants sont définis dans *Interfaces géométriques* : [\[GÉOMÉTRIE\]](#)

- [DOMMatrix](#) interface et [élément m11](#) , [élément m12](#) , [élément m21](#) , [élément m22](#) , [élément m41](#) et [élément m42 associés](#)
- [DOMMatrix2DInit](#) et [DOMMatrixInit](#) dictionnaires
- Les algorithmes [créer un DOMMatrix à partir d'un dictionnaire](#) et [créer un DOMMatrix à partir d'un dictionnaire 2D](#) pour [DOMMatrix2DInit](#) ou [DOMMatrixInit](#)
- Le [DOMPointInit](#) dictionnaire et les membres **x** et **y** associés

Les termes suivants sont définis dans le *CSS Scoping* : [\[CSSSCOPING\]](#)

- [arbre plat](#)

Les termes et fonctionnalités suivants sont définis dans *CSS Color Adjustment* : [\[CSSCOLORADJUST\]](#)

- ['Schéma de couleur'](#)
- [schémas de couleurs pris en charge par la page](#)

Le terme suivant est défini dans *CSS Pseudo-Elements* : [\[CSSPSEUDO\]](#)

- ['::bouton-sélecteur-de-fichier'](#)

Les termes suivants sont définis dans *CSS Containment* : [\[CSSCONTAIN\]](#)

- [ignore son contenu](#)
- [confinement de mise en page](#)

## Observateur d'intersection

Le terme suivant est défini dans *Intersection Observer* : [\[INTERSECTIONOBSERVER\]](#)

- [exécuter les étapes de mise à jour des observations d'intersection](#)
- [IntersectionObserver](#)
- [IntersectionObserverInit](#)
- [observe](#)
- [unobserve](#)
- [isIntersecting](#)
- [target](#)

## Redimensionner l'observateur

Les termes suivants sont définis dans *Resize Observer* : [\[RESIZEOBSERVER\]](#)

- [recueillir des observations de redimensionnement actif en profondeur](#)
- [a des observations de redimensionnement actives](#)
- [a ignoré les observations de redimensionnement](#)
- [diffuser des observations de redimensionnement actives](#)
- [livrer une erreur de boucle de redimensionnement](#)

## WebGL

Les interfaces suivantes sont définies dans les spécifications WebGL : [\[WEBGL\]](#)

- [WebGLRenderingContext](#) interface
- [WebGL2RenderingContext](#) interface
- [WebGLContextAttributes](#) dictionnaire

## WebGPU

Les interfaces suivantes sont définies dans *WebGPU* : [\[WEBGPU\]](#)

- [GPUCanvasContext](#) interface

## WebVTT

Les implémentations peuvent prendre en charge WebVTT en tant que format de piste de texte pour les sous-titres, les légendes, les métadonnées, etc., pour les ressources multimédias. [\[WEBVTT\]](#)

Les termes suivants, utilisés dans cette spécification, sont définis dans *WebVTT* :

- [Fichier WebVTT](#)
- [Fichier WebVTT utilisant le texte de repère](#)
- [Fichier WebVTT utilisant uniquement des repères imbriqués](#)
- [Analyseur WebVTT](#)
- Les [règles de mise à jour de l'affichage des pistes texte WebVTT](#)
- [La direction d'écriture de la piste de texte](#) WebVTT
- [VTT Cue](#) interface

## ARIA

L' **role** attribut est défini dans *Accessible Rich Internet Applications* ( *ARIA* ), tout comme les rôles suivants : [\[ARIA\]](#)

- [button](#)
- [presentation](#)

De plus, les **aria-\***attributs de contenu suivants sont définis dans *ARIA* : [\[ARIA\]](#)

- [aria-checked](#)
- [aria-describedby](#)
- [aria-disabled](#)
- [aria-label](#)

Enfin, les termes suivants sont définis *ARIA* : [\[ARIA\]](#)

- [rôle](#)
- [nom accessible](#)
- L' [ARIAMixin](#) interface, avec ses [ARIAMixin étapes getter](#) associées et ses hooks [ARIAMixin d'étapes setter](#)

## Politique de sécurité du contenu

Les termes suivants sont définis dans *la politique de sécurité du contenu* : [\[CSP\]](#)

- [Politique de sécurité du contenu](#)
- [disposition](#)
- [ensemble de directives](#)
- [Directive relative à la politique de sécurité du contenu](#)
- [Liste CSP](#)
- La [syntaxe de la politique de sécurité du contenu](#)
- [appliquer la politique](#)
- L' analyse d'un algorithme [de politique de sécurité de contenu sérialisé](#)
- L' [initialisation Run CSP pour un](#) algorithme de document
- L' [initialisation Run CSP pour un](#) algorithme d'objet global
- Le [comportement en ligne de l'élément doit-il être bloqué par la politique de sécurité du contenu ?](#) algorithme
- La [demande de navigation de type doit-elle être bloquée par la politique de sécurité du contenu ?](#) algorithme
- La [réponse de navigation à la demande de navigation de type dans la cible doit-elle être bloquée par la politique de sécurité du contenu ?](#) algorithme
- La [report-uri directive](#)
- L' opération abstraite [EnsureCSPDoesNotBlockStringCompilation](#)
- La [base est-elle autorisée pour le document ?](#) algorithme
- La [frame-ancestors directive](#)
- La [sandbox directive](#)
- Le [contient une](#) propriété Content Security Policy fournie par l'en-tête.
- L' algorithme [Analyser les politiques de sécurité du contenu d'une réponse](#) .
- [SecurityPolicyViolationEvent](#) interface
- L' [securitypolicyviolation](#) événement

## Travailleurs des services

Les termes suivants sont définis dans *Service Workers* : [\[SW\]](#)

- [travailleur actif](#)
- [file d'attente de messages client](#)
- [contrôle](#)
- [gérer la récupération](#)
- [enregistrement des travailleurs du service de correspondance](#)
- [travailleur de service](#)
- [client travailleur de service](#)
- [ServiceWorker](#) interface
- [ServiceWorkerContainer](#) interface
- [ServiceWorkerGlobalScope](#) interface

## Contextes sécurisés

Les algorithmes suivants sont définis dans *les contextes sécurisés* : [\[SECURE-CONTEXTS\]](#)



- [L'URL est-elle potentiellement fiable ?](#)

## Politique d'autorisations

Les termes suivants sont définis dans *la politique d'autorisation* : [\[PERMISSIONSPOLICY\]](#)

- [politique d'autorisations](#)
- [fonctionnalité contrôlée par une stratégie](#)
- [politique de conteneur](#)
- [stratégie d'autorisations sérialisées](#)
- [liste d'autorisation par défaut](#)
- La [création d'un](#) algorithme de politique d'autorisations
- La [création d'une politique d'autorisations à partir d'un](#) algorithme de réponse
- La [fonctionnalité est activée par la stratégie pour](#) l'algorithme d'origine
- L' algorithme [des attributs de politique d'autorisations de processus](#)

## API de demande de paiement

La fonctionnalité suivante est définie dans *l'API de demande de paiement* : [\[PAYMENTREQUEST\]](#)

- [PaymentRequest](#) interface

## MathML

Bien que la prise en charge de MathML dans son ensemble ne soit pas requise par cette spécification (bien qu'elle soit encouragée, du moins pour les navigateurs Web), certaines fonctionnalités dépendent de l'implémentation de petites parties de MathML. [\[MATHML\]](#)

Les fonctionnalités suivantes sont définies dans *Mathematical Markup Language ( MathML )*:

- [annotation-xml](#) Élément [MathML](#)
- [math](#) Élément [MathML](#)
- [merror](#) Élément [MathML](#)
- [mi](#) Élément [MathML](#)
- [mn](#) Élément [MathML](#)
- [mo](#) Élément [MathML](#)
- [ms](#) Élément [MathML](#)
- [mtext](#) Élément [MathML](#)

## SVG

Bien que la prise en charge de SVG dans son ensemble ne soit pas requise par cette spécification (bien qu'elle soit encouragée, du moins pour les navigateurs Web), certaines fonctionnalités dépendent de l'implémentation de certaines parties de SVG.

Les agents utilisateurs qui implémentent SVG doivent implémenter la spécification SVG 2 , et non les révisions précédentes.

Les fonctionnalités suivantes sont définies dans la spécification SVG 2 : [\[SVG\]](#)

- [SVGElement](#)interface
- [SVGImageElement](#)interface
- [SVGScriptElement](#)interface
- [SVGSVGElement](#)interface
- [a](#)Élément [SVG](#)
- [desc](#)Élément [SVG](#)
- [foreignObject](#)Élément [SVG](#)
- [image](#)Élément [SVG](#)
- [script](#)Élément [SVG](#)
- [svg](#)Élément [SVG](#)
- [title](#)Élément [SVG](#)
- [use](#)Élément [SVG](#)
- [text-rendering](#)Propriété [SVG](#)

### Effets de filtre

Les fonctionnalités suivantes sont définies dans *les effets de filtre* : [\[FILTRES\]](#)

- [<liste-valeur-filtre>](#)

### Composition

Les fonctionnalités suivantes sont définies dans *Compositing et Blending* : [\[COMPOSITE\]](#)

- [<mode fusion>](#)
- [<mode composite>](#)
- [source sur](#)
- [copie](#)

### Planification coopérative des tâches en arrière-plan

Les fonctionnalités suivantes sont définies dans *Cooperative Scheduling of Background Tasks* : [\[REQUESTIDLECALLBACK\]](#)

- [requestIdleCallback\(\)](#)
- [démarrer un algorithme de période d'inactivité](#)

### Orientation de l'écran

Les termes suivants sont définis dans *l'orientation de l'écran* : [\[SCREENORIENTATION\]](#)

- [étapes de changement d'orientation de l'écran](#)

### Stockage

Les termes suivants sont définis dans *Stockage* : [\[STOCKAGE\]](#)

- [obtenir une carte des bouteilles de stockage local](#)

- [obtenir une carte de bouteille de stockage de session](#)
- [obtenir une clé de stockage à des fins autres que de stockage](#)
- [clé de stockage égale](#)
- [carte proxy de stockage](#)
- [clone hérité d'un hangar de stockage traversable](#)

## Manifeste d'application Web

Les fonctionnalités suivantes sont définies dans *Web App Manifest* : [\[MANIFEST\]](#)

- [manifeste d'application](#)
- [application web installée](#)
- [traiter le manifeste](#)

## Codecs Web

Les fonctionnalités suivantes sont définies dans *les WebCodecs* : [\[WEBCODECS\]](#)

- [VideoFrame](#) interface.
- [\[\[largeur d'affichage\]\]](#)
- [\[\[hauteur d'affichage\]\]](#)

## WebDriver

Les termes suivants sont définis dans *WebDriver* : [\[WEBDRIVER\]](#)

- [commande d'extension](#)
- [étapes finales à distance](#)
- [Erreur de pilote Web](#)
- [Code d'erreur du pilote Web](#)
- [argument invalide](#)
- [obtenir une propriété](#)
- [succès](#)
- [Considérations de sécurité de WebDriver](#)
- [contexte de navigation actuel](#)

## WebDriver BiDi

Les termes suivants sont définis dans *WebDriver BiDi* : [\[WEBDRIVERBIDI\]](#)

- [Statut de navigation WebDriver BiDi](#)
- [ID d'état de navigation](#)
- [état de navigation état](#)
- [statut de navigation annulé](#)
- [statut de navigation en attente](#)
- [état de navigation terminé](#)
- [URL d'état de navigation](#)
- [La navigation WebDriver BiDi a commencé](#)
- [Navigation WebDriver BiDi abandonnée](#)
- [Échec de la navigation WebDriver BiDi](#)
- [Le téléchargement de WebDriver BiDi a commencé](#)
- [Fragment WebDriver BiDi navigué](#)

- [Contenu WebDriver BiDi DOM chargé](#)
- [Chargement WebDriver BiDi terminé](#)
- [Invite utilisateur WebDriver BiDi fermée](#)
- [Invite utilisateur WebDriver BiDi ouverte](#)

## API de cryptographie Web

Les termes suivants sont définis dans *l'API de cryptographie Web* : [\[WEBCRYPTO\]](#)

- [générer un UUID aléatoire](#)

## WebSockets

Les termes suivants sont définis dans *WebSockets* : [\[WEBSOCKETS\]](#)

- [WebSocket](#)
- [faire disparaître](#)

## Authentification Web : une API pour accéder aux informations d'identification de la clé publique

Les termes suivants sont définis dans *l'authentification Web : Une API pour accéder aux identifiants de clé publique* : [\[WEBAUTHN\]](#)

- [identifiant de clé publique](#)

## Gestion des informations d'identification

Les termes suivants sont définis dans *Credential Management* : [\[CREDMAN\]](#)

- [médiation conditionnelle](#)
- [justificatif](#)
- [navigator.credentials.get\(\)](#)

## Console

Les termes suivants sont définis dans *la console* : [\[CONSOLE\]](#)

- [signaler un avertissement à la console](#)

Cette spécification ne *nécessite* pas la prise en charge d'un protocole réseau particulier, d'un langage de feuille de style, d'un langage de script ou de l'une des spécifications DOM autres que celles requises dans la liste ci-dessus. Cependant, le langage décrit par cette spécification est biaisé vers CSS comme langage de style, JavaScript comme langage de script et HTTP comme protocole réseau, et plusieurs fonctionnalités supposent que ces langages et protocoles sont utilisés.

Un agent utilisateur qui implémente le protocole HTTP doit également implémenter *le mécanisme de gestion d'état HTTP (cookies)*. [\[HTTP\]](#) [\[COOKIES\]](#)

*Cette spécification peut avoir certaines exigences supplémentaires sur les codages de caractères, les formats d'image, les formats audio et les formats vidéo dans les sections respectives.*

## 2.1.10 Extensibilité

Les extensions d'agent utilisateur propriétaires spécifiques au fournisseur de cette spécification sont fortement déconseillées. Les documents ne doivent pas utiliser de telles extensions, car cela réduit l'interopérabilité et fragmente la base d'utilisateurs, permettant uniquement aux utilisateurs d'agents utilisateurs spécifiques d'accéder au contenu en question.

Toutes les extensions doivent être définies de manière à ce que l'utilisation d'extensions ne contredise ni ne provoque la non-conformité des fonctionnalités définies dans la spécification.

Par exemple, bien qu'il soit fortement déconseillé de le faire, une implémentation pourrait ajouter un nouvel attribut IDL " `typeTime` " à un contrôle qui renverrait le temps qu'il a fallu à l'utilisateur pour sélectionner la valeur actuelle d'un contrôle (par exemple). D'autre part, définir un nouveau contrôle qui apparaît dans le elements tableau d'un formulaire serait en violation de l'exigence ci-dessus, car cela violerait la définition de elements donnée dans cette spécification.

---

Lorsque des extensions indépendantes du fournisseur de cette spécification sont nécessaires, soit cette spécification peut être mise à jour en conséquence, soit une spécification d'extension peut être écrite qui remplace les exigences de cette spécification. Lorsqu'une personne appliquant cette spécification à ses activités décide qu'elle reconnaîtra les exigences d'une telle spécification d'extension, celle-ci devient une **spécification applicable** aux fins des exigences de conformité de cette spécification.

*Quelqu'un pourrait écrire une spécification qui définit n'importe quel flux d'octets arbitraires comme conforme, puis prétendre que ses déchets aléatoires sont conformes. Cependant, cela ne signifie pas que leur bric-à-brac aléatoire est réellement conforme aux besoins de tout le monde : si quelqu'un d'autre décide que cette spécification ne s'applique pas à son travail, alors il peut légitimement dire que le bric-à-brac susmentionné n'est que cela, du bric-à-brac, et non conforme du tout. En ce qui concerne la conformité, ce qui compte dans une communauté particulière, c'est ce que cette communauté convient d'appliquer.*

---

Les agents utilisateurs doivent traiter les éléments et attributs qu'ils ne comprennent pas comme sémantiquement neutres ; en les laissant dans le DOM (pour les processeurs DOM) et en les stylisant selon CSS (pour les processeurs CSS), mais sans en déduire aucune signification.

Lorsque la prise en charge d'une fonctionnalité est désactivée (par exemple, en tant que mesure d'urgence pour atténuer un problème de sécurité, ou pour aider au développement, ou pour des raisons de performances), les agents utilisateurs doivent agir comme s'ils n'avaient aucune prise en charge de la fonctionnalité, et comme si la fonction n'a pas été mentionnée dans cette spécification. Par exemple, si une fonctionnalité particulière est accessible via un attribut dans une interface Web IDL, l'attribut lui-même serait omis des objets qui implémentent cette interface - laisser l'attribut sur l'objet mais le faire retourner null ou lever une exception est insuffisant.

### 2.1.11 Interactions avec XPath et XSLT

Les implémentations de XPath 1.0 qui fonctionnent sur [des documents HTML](#) analysés ou créés de la manière décrite dans cette spécification (par exemple dans le cadre de l' `document.evaluate()` API) doivent agir comme si la modification suivante était appliquée à la spécification XPath 1.0.

Tout d'abord, supprimez ce paragraphe :

*Un [QName](#) dans le test de nœud est développé en un [nom développé](#) à l'aide des déclarations d'espace de noms du contexte d'expression. C'est de la même manière que l'expansion est effectuée pour les noms de type d'élément dans les balises de début et de fin, sauf que l'espace de noms par défaut déclaré avec n'est pas utilisé : si le `xmlns` QName [n'a](#) pas de préfixe, alors l'URI de l'espace de noms est nul (c'est de la même manière les noms d'attributs sont développés). C'est une erreur si le [QName](#) a un préfixe pour lequel il n'y a pas de déclaration d'espace de noms dans le contexte de l'expression.*

Ensuite, insérez à sa place ce qui suit :

*Un QName dans le test de nœud est développé en un nom développé à l'aide des déclarations d'espace de noms du contexte d'expression. Si le QName a un préfixe, il doit y avoir une déclaration d'espace de noms pour ce préfixe dans le contexte de l'expression, et l'URI d'espace de noms correspondant est celui qui est associé à ce préfixe. C'est une erreur si le QName a un préfixe pour lequel il n'y a pas de déclaration d'espace de noms dans le contexte de l'expression.*

Si le QName n'a pas de préfixe et que le type de nœud principal de l'axe est element, l'espace de noms d'élément par défaut est utilisé. Sinon, si le QName n'a pas de préfixe, l'URI de l'espace de noms est nul. L'espace de noms d'élément par défaut est un membre du contexte de l'expression XPath. La valeur de l'espace de noms d'élément par défaut lors de l'exécution d'une expression XPath via l'API XPath DOM3 est déterminée de la manière suivante :

1. Si le nœud de contexte provient d'un DOM HTML, l'espace de noms d'élément par défaut est "http://www.w3.org/1999/xhtml".
2. Sinon, l'URI de l'espace de noms d'élément par défaut est null.

Cela équivaut à ajouter la fonctionnalité d'espace de noms d'élément par défaut de XPath 2.0 à XPath 1.0 et à utiliser l'espace de noms HTML comme espace de noms d'élément par défaut pour les documents HTML. Il est motivé par le désir que les implémentations soient compatibles avec le contenu HTML hérité tout en prenant en charge les changements que cette spécification introduit dans HTML concernant l'espace de noms utilisé pour les éléments HTML, et par le désir d'utiliser XPath 1.0 plutôt que XPath 2.0.

Cette modification est une [violation délibérée](#) de la spécification XPath 1.0, motivée par le désir que les implémentations soient compatibles avec le contenu hérité tout en prenant en charge les modifications que cette spécification introduit dans HTML concernant l'espace de noms utilisé pour les éléments HTML. [\[XPATH10\]](#)

---

Les processeurs XSLT 1.0 sortant vers un DOM lorsque la méthode de sortie est « html » (soit explicitement, soit via la règle par défaut dans XSLT 1.0) sont affectés comme suit :

Si le programme de transformation génère un élément sans espace de noms, le processeur doit, avant de construire le nœud d'élément DOM correspondant, changer l'espace de noms de l'élément en espace de noms HTML , ASCII-minuscules le nom [local](#) de l'élément et [ASCII-minuscules](#) les noms de tous les attributs sans espace de noms sur l'élément.

Cette exigence est une [violation délibérée](#) de la spécification XSLT 1.0, nécessaire car cette spécification modifie les espaces de noms et les règles de sensibilité à la casse de HTML d'une manière qui serait autrement incompatible avec les transformations XSLT basées sur DOM. (Les processeurs qui sérialisent la sortie ne sont pas affectés.) [\[XSLT10\]](#)

---



Cette spécification ne spécifie pas précisément comment le traitement XSLT interagit avec l'infrastructure [de l'analyseur HTML](#) (par exemple, si un processeur XSLT agit comme s'il plaçait des éléments dans une [pile d'éléments ouverts](#) ). Cependant, les processeurs XSLT doivent [arrêter l'analyse](#) s'ils se terminent avec succès et doivent d'abord [mettre à jour l'état de préparation du document actuel](#) sur " `interactive`", puis sur " `complete`" s'ils sont abandonnés.

---

Cette spécification ne précise pas comment XSLT interagit avec l'algorithme [de navigation](#) , comment il s'intègre dans la [boucle d'événements](#) , ni comment les pages d'erreur doivent être gérées (par exemple, si les erreurs XSLT doivent remplacer une sortie XSLT incrémentielle, ou sont rendues en ligne, etc. ).

*Il existe également des commentaires non normatifs supplémentaires concernant l'interaction de XSLT et HTML [dans la script section élément](#) , et de XSLT, XPath et HTML [dans la template section élément](#) .*

## 2.2 Fonctionnalités contrôlées par des politiques

Ce document définit les [fonctionnalités contrôlées par des règles](#) suivantes :

### MDN

- " `autoplay`", dont la liste d'autorisation par [défaut](#) `'self'` est .
- " `cross-origin-isolated`", dont la liste d'autorisation par [défaut](#) `'self'` est .

## 2.3 Microsyntaxes courantes

Il existe différents endroits dans HTML qui acceptent des types de données particuliers, tels que des dates ou des nombres. Cette section décrit les critères de conformité pour le contenu dans ces formats et comment les analyser.

*Les implémenteurs sont fortement invités à examiner attentivement toutes les bibliothèques tierces qu'ils pourraient envisager d'utiliser pour implémenter l'analyse des syntaxes décrites ci-dessous. Par exemple, les bibliothèques de dates sont susceptibles d'implémenter un comportement de gestion des erreurs qui diffère de ce qui est requis dans cette spécification, puisque le comportement de gestion des erreurs n'est souvent pas défini dans les spécifications qui décrivent des syntaxes de*



*date similaires à celles utilisées dans cette spécification, et donc les implémentations ont tendance à varier considérablement dans la façon dont ils gèrent les erreurs.*

### 2.3.1 Idiomes d'analyseur courants

Certains des micro-analyseurs décrits ci-dessous suivent le modèle consistant à avoir une variable *d'entrée* qui contient la chaîne analysée et une variable de *position* pointant vers le caractère suivant à analyser dans *input*.

### 2.3.2 Attributs booléens

Un certain nombre d'attributs sont **des attributs booléens**. La présence d'un attribut booléen sur un élément représente la vraie valeur, et l'absence de l'attribut représente la fausse valeur.

Si l'attribut est présent, sa valeur doit être soit la chaîne vide, soit une valeur qui est une correspondance [ASCII insensible à la casse](#) pour le nom canonique de l'attribut, sans espace blanc de début ou de fin.

*Les valeurs "true" et "false" ne sont pas autorisées sur les attributs booléens. Pour représenter une valeur fausse, l'attribut doit être complètement omis.*

Voici un exemple de case cochée et désactivée. Les attributs `checked` et `disabled` sont les attributs booléens.

```
<label><input type=checkbox checked name=cheese disabled>
Cheese</label>
```

Cela pourrait être écrit de manière équivalente comme ceci:

```
<label><input type=checkbox checked=checked name=cheese
disabled=disabled> Cheese</label>
```

Vous pouvez également mélanger les styles ; ce qui suit est toujours équivalent :

```
<label><input type='checkbox' checked name=cheese disabled="">
Cheese</label>
```

### 2.3.3 Mots clés et attributs énumérés

Certains attributs, appelés **attributs énumérés**, prennent un ensemble fini d'états. L'état d'un tel attribut est dérivé en combinant la valeur de l'attribut, un

ensemble de correspondances mot-clé/état donné dans la spécification de chaque attribut, et deux états spéciaux possibles qui peuvent également être donnés dans la spécification de l'attribut. Ces états spéciaux sont la **valeur invalide par défaut** et la **valeur manquante par défaut**.

*Plusieurs mots-clés peuvent correspondre au même état.  
La chaîne vide peut être un mot-clé valide. Notez que la [valeur par défaut de la valeur manquante](#) s'applique uniquement lorsque l'attribut est missing, pas lorsqu'il est présent avec une valeur de chaîne vide.*

Pour déterminer l'état d'un attribut, procédez comme suit :

1. Si l'attribut n'est pas spécifié :
  1. Si l'attribut a un état [par défaut de valeur manquante](#) défini, renvoyez cet état [par défaut de valeur manquante](#).
  2. Sinon, ne renvoie aucun état.
2. Si la valeur de l'attribut est une correspondance [ASCII non sensible à la casse](#) pour l'un des mots-clés définis pour l'attribut, renvoie l'état représenté par ce mot-clé.
3. Si l'attribut a un état [par défaut de valeur invalide](#) défini, renvoyez cet état [par défaut de valeur invalide](#).
4. Ne renvoie aucun état.

Pour des raisons de conformité de création, si un attribut énuméré est spécifié, la valeur de l'attribut doit être une correspondance [ASCII insensible à la casse](#) pour l'un des mots-clés conformes pour cet attribut, sans espace de début ou de fin.

À des fins [de réflexion](#), les états qui ont des mots-clés qui leur correspondent sont dits avoir un **mot-clé canonique**. Celui-ci est déterminé comme suit :

- S'il n'y a qu'un seul mot-clé mappé à l'état donné, alors c'est ce mot-clé.
- S'il n'y a qu'un seul mappage de mot-clé *conforme* à l'état donné, alors c'est ce mot-clé conforme.
- Sinon, le mot-clé canonique de l'état sera explicitement donné dans la spécification de l'attribut.

## 2.3.4 Numéros

### 2.3.4.1 Entiers signés

Une chaîne est un **entier valide** si elle se compose d'un ou plusieurs [chiffres ASCII](#) , éventuellement précédés d'un caractère U+002D HYPHEN-MINUS (-).

Un [nombre entier valide](#) sans préfixe U+002D Trait d'union-Moins (-) représente le nombre représenté en base dix par cette chaîne de chiffres. Un [entier valide](#) avec un préfixe U+002D TRAIN-MOINS (-) représente le nombre représenté en base dix par la chaîne de chiffres qui suit le U+002D TRAIN-MOINS, soustrait de zéro.

Les **règles d'analyse des entiers** sont données dans l'algorithme suivant. Lorsqu'elles sont invoquées, les étapes doivent être suivies dans l'ordre indiqué, en abandonnant à la première étape qui renvoie une valeur. Cet algorithme renverra soit un entier soit une erreur.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. Soit *signe* la valeur "positive".
4. [Ignorer les espaces blancs ASCII](#) dans *la position d'entrée* donnée .
5. Si *position* dépasse la fin de *input* , renvoie une erreur.
6. Si le caractère indiqué par *la position* (le premier caractère) est un caractère U+002D HYPHEN-MOINS (-) :

1. Soit *signe* "négatif".
2. *Position* d'avance au caractère suivant.
3. Si *position* dépasse la fin de *input* , renvoie une erreur.

Sinon, si le caractère indiqué par *la position* (le premier caractère) est un caractère U+002B PLUS SIGN (+) :

4. *Position* d'avance au caractère suivant. (Le " +" est ignoré, mais il n'est pas conforme.)
5. Si *position* dépasse la fin de *input* , renvoie une erreur.
7. Si le caractère indiqué par *position* n'est pas un [chiffre ASCII](#) , renvoie une erreur.
8. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée* et interprétez la séquence résultante comme un entier de base dix. Soit *value* cet entier.
9. Si *signe* est "positif", renvoie *value* , sinon renvoie le résultat de la soustraction *de value* à zéro.

#### 2.3.4.2 Entiers non négatifs

Une chaîne est un **entier non négatif valide** si elle se compose d'un ou plusieurs [chiffres ASCII](#) .

Un [entier non négatif valide](#) représente le nombre représenté en base dix par cette chaîne de chiffres.

Les **règles d'analyse des entiers non négatifs** sont données dans l'algorithme suivant. Lorsqu'elles sont invoquées, les étapes doivent être suivies dans l'ordre indiqué, en abandonnant à la première étape qui renvoie une valeur. Cet algorithme renverra soit zéro, soit un entier positif, soit une erreur.

1. Soit *input* la chaîne analysée.
2. Soit *value* le résultat de l'analyse de l'entrée en utilisant les [règles d'analyse des entiers](#) .
3. Si *la valeur* est une erreur, renvoie une erreur.
4. Si *la valeur* est inférieure à zéro, renvoie une erreur.
5. *Valeur* de retour .

#### 2.3.4.3 Nombres à virgule flottante

Une chaîne est un **nombre à virgule flottante valide** si elle se compose de :

1. Facultativement, un caractère U+002D TRAIN D'ASSEMBLAGE-MOINS (-).
2. Un ou les deux éléments suivants, dans l'ordre indiqué :
  1. Une série d'un ou plusieurs [chiffres ASCII](#) .
  2. Les deux éléments suivants, dans l'ordre indiqué :
    1. Un seul caractère U+002E FULL STOP (.).
    2. Une série d'un ou plusieurs [chiffres ASCII](#) .
3. Facultativement :
  1. Soit un caractère U+0065 LETTRE MINUSCULE LATINE E (e) ou un caractère U+0045 LETTRE MAJUSCULE LATINE E (E).
  2. Facultativement, un caractère U+002D HYPHEN-MOINS (-) ou U+002B PLUS SIGN (+).
  3. Une série d'un ou plusieurs [chiffres ASCII](#) .

Un [nombre à virgule flottante valide](#) représente le nombre obtenu en multipliant le significande par dix élevé à la puissance de l'exposant, où le significande est le premier nombre, interprété en base dix (y compris le point décimal et le nombre après le point décimal, si quelconque, et interprétant le significande comme un nombre négatif si toute la chaîne commence par un caractère U+002D HYPHEN-MOINS (-) et que le nombre n'est pas zéro), et où l'exposant est le nombre après le

E, le cas échéant (interprété comme un nombre négatif s'il y a un caractère U+002D HYPHEN-MOINS (-) entre le E et le nombre et que le nombre n'est pas zéro, ou bien en ignorant un caractère U+002B PLUS SIGN (+) entre le E et le nombre Si il y en a un). S'il n'y a pas de E, alors l'exposant est traité comme zéro.

*Les valeurs Infinity et Not-a-Number (NaN) ne sont pas des nombres à virgule flottante valides .*

*Le concept de nombre à virgule flottante valide n'est généralement utilisé que pour restreindre ce qui est autorisé pour les auteurs, tandis que les exigences de l'agent utilisateur utilisent les règles d'analyse des valeurs de nombre à virgule flottante ci-dessous (par exemple, l'maxattribut de l'progressélément). Cependant, dans certains cas, les exigences de l'agent utilisateur incluent la vérification si une chaîne est un nombre à virgule flottante valide (par exemple, l' algorithme de nettoyage de valeur pour l' état Number de l' inputélément, ou l' algorithme d'analyse d'un attribut srcset ).*

La **meilleure représentation du nombre  $n$  sous forme de nombre à virgule flottante** est la chaîne obtenue en exécutant ToString (  $n$  ). L'opération abstraite ToString n'est pas déterminée de manière unique. Lorsqu'il existe plusieurs chaînes possibles pouvant être obtenues à partir de ToString pour une valeur particulière, l'agent utilisateur doit toujours renvoyer la même chaîne pour cette valeur (bien qu'elle puisse différer de la valeur utilisée par d'autres agents utilisateurs).

Les **règles d'analyse des valeurs de nombres à virgule flottante** sont données dans l'algorithme suivant. Cet algorithme doit être abandonné à la première étape qui renvoie quelque chose. Cet algorithme renverra soit un nombre, soit une erreur.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. Soit *value* la valeur 1.
4. Soit *diviseur* la valeur 1.
5. Soit *exposant* la valeur 1.
6. Ignorer les espaces blancs ASCII dans *la position d'entrée* donnée .
7. Si *position* dépasse la fin de *input* , renvoie une erreur.
8. Si le caractère indiqué par *la position* est un caractère U+002D TYPE-MOINS (-) :
  1. Remplacez *la valeur* et *le diviseur* par -1.
  2. *Position* d'avance au caractère suivant.
  3. Si *position* dépasse la fin de *input* , renvoie une erreur.

Sinon, si le caractère indiqué par *la position* (le premier caractère) est un caractère U+002B PLUS SIGN (+) :

4. *Position* d'avance au caractère suivant. (Le " + " est ignoré, mais il n'est pas conforme.)
5. Si *position* dépasse la fin de *input* , renvoie une erreur.
9. Si le caractère indiqué par *la position* est un U+002E FULL STOP (.), et qu'il ne s'agit pas du dernier caractère de *l'entrée* , et que le caractère après le caractère indiqué par *la position* est un [chiffre ASCII](#) , réglez *la valeur* sur zéro et passez au étape *fraction* étiquetée .
10. Si le caractère indiqué par *position* n'est pas un [chiffre ASCII](#) , renvoie une erreur.
11. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée* et interprétez la séquence résultante comme un entier de base dix. Multipliez *la valeur* par cet entier.
12. Si *la position* dépasse la fin de *l'entrée* , passez à l'étape intitulée *conversion* .
13. *Fraction* : Si le caractère indiqué par *la position* est un U+002E FULL STOP (.), exécutez ces sous-étapes :
  1. *Position* d'avance au caractère suivant.
  2. Si *la position* dépasse la fin de *la saisie* , ou si le caractère indiqué par *la position* n'est pas un [chiffre ASCII](#) , U+0065 LETTRE MINUSCULE LATINE E (e), ou U+0045 LETTRE MAJUSCULE LATINE E (E), alors sautez à l'étape *conversion* étiquetée .
  3. Si le caractère indiqué par *la position* est un caractère U+0065 LETTRE MINUSCULE LATINE E (e) ou un caractère U+0045 LETTRE MAJUSCULE LATINE E (E), ignorez le reste de ces sous-étapes.
  4. *Boucle de fraction* : Multipliez *le diviseur* par dix.
  5. Ajoutez la valeur du caractère indiqué par *position* , interprété comme un chiffre de base dix (0..9) et divisé par *diviseur* , à *value* .
  6. *Position* d'avance au caractère suivant.
  7. Si *la position* dépasse la fin de *l'entrée* , passez à l'étape intitulée *conversion* .
  8. Si le caractère indiqué par *la position* est un [chiffre ASCII](#) , revenez à l'étape étiquetée *boucle de fraction* dans ces sous-étapes.
14. Si le caractère indiqué par *position* est U+0065 (e) ou un U+0045 (E), alors :
  1. *Position* d'avance au caractère suivant.

2. Si *la position* dépasse la fin de *l'entrée* , passez à l'étape intitulée *conversion* .
3. Si le caractère indiqué par *la position* est un caractère U+002D TYPE-MOINS (-) :

1. Changer *l'exposant* en -1.
2. *Position* d'avance au caractère suivant.
3. Si *la position* dépasse la fin de *l'entrée* , passez à l'étape intitulée *conversion* .

Sinon, si le caractère indiqué par *position* est un caractère U+002B PLUS SIGN (+) :

4. *Position* d'avance au caractère suivant.
5. Si *la position* dépasse la fin de *l'entrée* , passez à l'étape intitulée *conversion* .
4. Si le caractère indiqué par *la position* n'est pas un [chiffre ASCII](#) , passez à l' étape intitulée *conversion* .
5. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en *entrée* et interprétez la séquence résultante comme un entier de base dix. Multipliez *l'exposant* par cet entier.
6. Multipliez *la valeur* par dix élevée à la puissance ème de l' *exposant* .

15. *Conversion* : Soit *S* l'ensemble des valeurs finies à virgule flottante double précision IEEE 754 sauf -0, mais avec deux valeurs spéciales ajoutées :  $2^{1024}$  et  $-2^{1024}$  .

16. Soit *valeur arrondie* le nombre dans *S* le plus proche de *valeur* , en sélectionnant le nombre avec un signifiant pair s'il existe deux valeurs également proches. (Les deux valeurs spéciales  $2^{1024}$  et  $-2^{1024}$  sont considérées comme ayant des significandes paires à cette fin.)

17. Si *la valeur arrondie* est  $2^{1024}$  ou  $-2^{1024}$  , renvoie une erreur.

18. Renvoie *valeur-arrondie* .

#### 2.3.4.4 Pourcentages et longueurs

Les **règles d'analyse des valeurs de dimension** sont données dans l'algorithme suivant. Lorsqu'elles sont invoquées, les étapes doivent être suivies dans l'ordre indiqué, en abandonnant à la première étape qui renvoie une valeur. Cet algorithme

renverra soit un nombre supérieur ou égal à 0,0, soit un échec ; si un nombre est renvoyé, il est ensuite catégorisé comme un pourcentage ou une longueur.

1. Soit *input* la chaîne analysée.
2. Soit *position* une [variable de position](#) pour *input* , pointant initialement au début de *input* .
3. [Ignorer les espaces blancs ASCII](#) dans *la position d'entrée* donnée .
4. Si *la position* est au-delà de la fin de *l'entrée* ou si le point de code à *la position* dans *l'entrée* n'est pas un [chiffre ASCII](#) , alors renvoie l'échec.
5. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée* et interprétez la séquence résultante comme un entier de base dix. Soit *value* ce nombre.
6. Si *position* est au-delà de la fin de *input* , alors renvoie *value* sous forme de longueur.
7. Si le point de code à *la position* dans *l'entrée* est U+002E (.), alors :
  1. Avancez *la position* de 1.
  2. Si *la position* dépasse la fin de *l'entrée* ou si le point de code à *la position* dans *l'entrée* n'est pas un [chiffre ASCII](#) , renvoie la [valeur de dimension actuelle](#) avec *valeur* , *entrée* et *position* .
  3. Soit *diviseur* la valeur 1.
  4. Alors que c'est vrai :
    1. Multipliez *le diviseur* par dix.
    2. Ajoutez la valeur du point de code à *la position* dans *l'entrée* , interprétée comme un chiffre de base dix (0..9) et divisée par *le diviseur* , à *la valeur* .
    3. Avancez *la position* de 1.
    4. Si *position* est au-delà de la fin de *input* , alors renvoie *value* sous forme de longueur.
    5. Si le point de code à *la position* dans *l'entrée* n'est pas un [chiffre ASCII](#) , alors [break](#) .
8. Renvoie la [valeur de dimension actuelle](#) avec *value* , *input* et *position* .

La **valeur de cote actuelle** , compte tenu de *la valeur* , de *l'entrée* et de *la position* , est déterminée comme suit :



1. Si *position* est au-delà de la fin de *input* , alors renvoie *value* sous forme de longueur.
2. Si le point de code à *la position* dans *l'entrée* est U+0025 (%), alors *la valeur* de retour sous forme de pourcentage.
3. *Valeur* de retour sous forme de longueur.

#### 2.3.4.5 Pourcentages et longueurs non nuls

Les **règles d'analyse des valeurs de dimension non nulles** sont données dans l'algorithme suivant. Lorsqu'elles sont invoquées, les étapes doivent être suivies dans l'ordre indiqué, en abandonnant à la première étape qui renvoie une valeur. Cet algorithme renverra soit un nombre supérieur à 0,0, soit une erreur ; si un nombre est renvoyé, il est ensuite catégorisé comme un pourcentage ou une longueur.

1. Soit *input* la chaîne analysée.
2. Soit *value* le résultat de l'analyse de *l'entrée* à l'aide des [règles d'analyse des valeurs de dimension](#) .
3. Si *la valeur* est une erreur, renvoie une erreur.
4. Si *la valeur* est zéro, renvoie une erreur.
5. Si *la valeur* est un pourcentage, renvoie *la valeur* sous forme de pourcentage.
6. *Valeur* de retour sous forme de longueur.

#### 2.3.4.6 Listes de nombres à virgule flottante

Une **liste valide de nombres à virgule flottante** est un nombre de [nombres à virgule flottante valides](#) séparés par des caractères U+002C COMMA, sans aucun autre caractère (par exemple, aucun [espace ASCII](#) ). De plus, il peut y avoir des restrictions sur le nombre de nombres à virgule flottante pouvant être donnés ou sur la plage de valeurs autorisées.

Les **règles d'analyse d'une liste de nombres à virgule flottante** sont les suivantes :

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.

3. Soit *nombres* une liste initialement vide de nombres à virgule flottante. Cette liste sera le résultat de cet algorithme.
4. [Collectez une séquence de points de code](#) qui sont des caractères [ASCII whitespace](#) , U+002C COMMA ou U+003B SEMICOLON à partir de *la position* donnée en entrée . Cela ignore tous les délimiteurs principaux.
5. Tant que *la position* n'est pas au-delà de la fin de *l'entrée* :
  1. [Collectez une séquence de points de code](#) qui ne sont pas [des espaces blancs ASCII](#) , des virgules U+002C, des points-virgules U+003B, [des chiffres ASCII](#) , des caractères U+002E FULL STOP ou U+002D HYPHEN-MINUS à partir de *la position* donnée en entrée . Cela saute au-delà des ordures principales.
  2. [Collectez une séquence de points de code](#) qui ne sont pas [des espaces blancs ASCII](#) , des caractères U+002C COMMA ou U+003B SEMICOLON à partir de *la position* donnée en entrée , et laissez *un nombre non analysé* être le résultat.
  3. Soit *nombre* le résultat de l'analyse d' *un nombre non analysé* à l'aide des [règles d'analyse des nombres à virgule flottante](#) .
  4. Si *le nombre* est une erreur, réglez *le nombre* sur zéro.
  5. Ajouter *un nombre* aux *nombres* .
  6. [Collectez une séquence de points de code](#) qui sont des caractères [ASCII whitespace](#) , U+002C COMMA ou U+003B SEMICOLON à partir de *la position* donnée en entrée . Cela saute le délimiteur.
6. *Numéros de retour* .

#### 2.3.4.7 Listes de dimensions

Les **règles d'analyse d'une liste de dimensions** sont les suivantes. Ces règles renvoient une liste de zéro ou plusieurs paires composées d'un nombre et d'une unité, l'unité étant l'une des *pourcentage* , *relatif* et *absolu* .

1. Soit *l'entrée brute* la chaîne analysée.
2. Si le dernier caractère de *l'entrée brute* est un caractère virgule U+002C (,), supprimez ce caractère de *l'entrée brute* .
3. [Divisez l' entrée brute de la chaîne par des virgules](#) . Soit *jetons bruts* la liste de jetons résultante.
4. Soit *result* une liste vide de paires nombre/unité.

5. Pour chaque jeton dans *jetons bruts* , exécutez les sous-étapes suivantes :

1. Soit *input* le jeton.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. Soit *valeur* le nombre 0.
4. Soit *unité* absolue . \_
5. Si *la position* dépasse la fin de *l'entrée* , réglez *l'unité* sur *relative* et passez à la dernière sous-étape.
6. Si le caractère à *la position* est un [chiffre ASCII](#) , [collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *l'entrée* donnée *position* , interprétez la séquence résultante comme un entier en base dix et incrémentez *la valeur* de cet entier.
7. Si le caractère à *la position* est U+002E (.), alors :
  1. [Collectez une séquence de points de code](#) composée d' [espaces ASCII](#) et [de chiffres ASCII](#) à partir de *la position* donnée en *entrée* . Soit *s* la suite résultante.
  2. Supprimez tous [les espaces blancs ASCII](#) dans *s* .
  3. Si *s* n'est pas la chaîne vide, alors :
    1. Soit *longueur* le nombre de caractères dans *s* (après suppression des espaces).
    2. Soit *fraction* le résultat de l'interprétation de *s* comme un entier de base dix, puis de la division de ce nombre par  $10^{length}$  .
    3. Incrémenter *la valeur* par *fraction* .
8. [Ignorer les espaces blancs ASCII](#) dans *la position d'entrée* donnée .
9. Si le caractère à *la position* est un caractère U+0025 SIGNE POUR CENT (%), alors réglez *l'unité* sur *pourcentage* .  
  
Sinon, si le caractère à *la position* est un caractère U+002A ASTÉRISQUE (\*), alors réglez *l'unité* sur *relative* .
10. Ajouter une entrée au *résultat* consistant en le nombre donné par *valeur* et l'unité donnée par *unité* .

6. Renvoie le *résultat* de la liste .

### 2.3.5 Dates et heures

Dans les algorithmes ci-dessous, le **nombre de jours dans le mois** *mois* de l'année *année* est : 31 si *le mois* est 1, 3, 5, 7, 8, 10 ou 12 ; 30 si *le mois* est 4, 6, 9 ou 11 ; 29 si *le mois* est 2 et l'année est un nombre divisible par 400, ou si l'année est un nombre divisible par 4 mais pas par 100 ; et 28 sinon. Cela prend en compte les années bissextiles du calendrier grégorien. [\[GRÉGORIEN\]](#)

Lorsque [des chiffres ASCII](#) sont utilisés dans les syntaxes de date et d'heure définies dans cette section, ils expriment des nombres en base dix.

*Alors que les formats décrits ici sont destinés à être des sous-ensembles des formats ISO8601 correspondants, cette spécification définit les règles d'analyse de manière beaucoup plus détaillée que ISO8601. Les implémenteurs sont donc encouragés à examiner attentivement toutes les bibliothèques d'analyse de date avant de les utiliser pour implémenter les règles d'analyse décrites ci-dessous ; Les bibliothèques ISO8601 peuvent ne pas analyser les dates et les heures exactement de la même manière. [\[ISO8601\]](#)*

Lorsque cette spécification fait référence au **calendrier grégorien proleptique**, cela signifie le calendrier grégorien moderne, extrapolé vers l'arrière à l'année 1. Une date dans le [calendrier grégorien proleptique](#), parfois explicitement appelée **date proleptique-grégorienne**, est celle qui est décrite à l'aide de ce calendrier. même si ce calendrier n'était pas utilisé au moment (ou au lieu) en question. [\[GRÉGORIEN\]](#)

*L'utilisation du calendrier grégorien comme format filaire dans cette spécification est un choix arbitraire résultant des préjugés culturels des personnes impliquées dans la décision. Voir également la section traitant [des formats de date, d'heure et de nombre dans les formulaires \(pour les auteurs\)](#), [les notes d'implémentation concernant la localisation des contrôles de formulaire](#) et l'[time](#)élément.*

#### 2.3.5.1 Mois

Un **mois** consiste en une [date proleptique-grégorienne](#) spécifique sans information de fuseau horaire et sans information de date au-delà d'un an et d'un mois. [\[GRÉGORIEN\]](#)

Une chaîne est une **chaîne de mois valide** représentant une année *année* et un mois *mois* si elle se compose des composants suivants dans l'ordre indiqué :

1. Quatre [chiffres ASCII](#) ou plus, représentant l'année, où *année* > 0
2. Un caractère U+002D TIRET-MOINS (-)
3. Deux [chiffres ASCII](#), représentant le mois *mois*, dans la plage  $1 \leq \text{mois} \leq 12$

Les règles pour **analyser une chaîne de mois** sont les suivantes. Cela renverra soit une année et un mois, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Analysez un mois](#) pour obtenir *year* et *month* . Si cela ne renvoie rien, échouez.
4. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
5. Renvoie *l'année* et *le mois* .

Les règles d' **analyse d'un mois composant** , étant donné une chaîne *d'entrée* et une *position* , sont les suivantes. Cela renverra soit un an et un mois, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en entrée . Si la séquence collectée ne comporte pas au moins quatre caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit l' *année* .
2. Si *l'année* n'est pas un nombre supérieur à zéro, échouez.
3. Si *la position* est au-delà de la fin de *l'entrée* ou si le caractère à *la position* n'est pas un caractère U+002D HYPHEN-MINUS, alors échoue. Sinon, déplacer *la position* vers l'avant d'un caractère.
4. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en entrée . Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit le *mois* .
5. Si *le mois* n'est pas un nombre compris entre  $1 \leq \textit{mois} \leq 12$ , alors échoue.
6. Renvoie *l'année* et *le mois* .

### 2.3.5.2 Dates

Une **date** consiste en une [date proleptique-grégorienne](#) spécifique sans information de fuseau horaire, composée d'une année, d'un mois et d'un jour. [\[GRÉGORIEN\]](#)

Une chaîne est une **chaîne de date valide** représentant une année *année* , mois *mois* et jour *jour* si elle comprend les composants suivants dans l'ordre indiqué :

1. Une [chaîne de mois valide](#) , représentant *l'année* et *le mois*
2. Un caractère U+002D TIRET-MOINS (-)
3. Deux [chiffres ASCII](#) , représentant *le jour* , dans la plage  $1 \leq \text{jour} \leq \text{maxjour}$  où *maxjour* est le [nombre de jours dans le mois mois et l'année année](#)

Les règles pour **analyser une chaîne de date** sont les suivantes. Cela renverra soit une date, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Analyser un composant de date](#) pour obtenir *l'année* , *le mois* et *le jour* . Si cela ne renvoie rien, échouez.
4. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
5. Soit *date* la date avec année *année* , mois *mois* et jour *jour* .
6. *Date* de retour .

Les règles d' **analyse d'un composant date** , étant donné une chaîne *d'entrée* et une *position* , sont les suivantes. Cela renverra soit une année, un mois et un jour, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. [Analysez un mois](#) pour obtenir *year* et *month* . Si cela ne renvoie rien, échouez.
2. Soit *maxday* le [nombre de jours du mois mois de l'année année](#) .
3. Si *la position* est au-delà de la fin de *l'entrée* ou si le caractère à *la position* n'est pas un caractère U+002D HYPHEN-MINUS, alors échoue. Sinon, déplacer *la position* vers l'avant d'un caractère.
4. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en *entrée* . Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit le *jour* .
5. Si *day* n'est pas un nombre compris entre  $1 \leq \text{day} \leq \text{maxday}$  , alors échoue.
6. Renvoie *l'année* , *le mois* et *le jour* .

### 2.3.5.3 Dates sans année

Une **date sans année** se compose d'un mois grégorien et d'un jour dans ce mois, mais sans année associée. [\[GRÉGORIEN\]](#)

Une chaîne est une **chaîne de date sans année valide** représentant un mois *mois* et un jour *jour* si elle se compose des composants suivants dans l'ordre indiqué :

1. Facultativement, deux caractères U+002D HYPHEN-MOINS (-)
2. Deux [chiffres ASCII](#) , représentant le mois *mois* , dans la plage  $1 \leq \text{mois} \leq 12$
3. Un caractère U+002D TIRET-MOINS (-)
4. Deux [chiffres ASCII](#) , représentant *day* , dans la plage  $1 \leq \text{day} \leq \text{maxday}$  où *maxday* est le [nombre de jours](#) dans le mois *mois* et toute année bissextile arbitraire (par exemple 4 ou 2000)

*En d'autres termes, si le mois est " 02 ", c'est-à-dire février, alors le jour peut être 29, comme si l'année était bissextile.*

Les règles pour **analyser une chaîne de date sans année** sont les suivantes. Cela renverra soit un mois et un jour, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Analysez un composant de date sans année](#) pour obtenir *le mois* et *le jour* . Si cela ne renvoie rien, échouez.
4. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
5. Renvoie *le mois* et *le jour* .

Les règles d' **analyse d'un composant de date sans année** , étant donné une chaîne *d'entrée* et une *position* , sont les suivantes. Cela renverra soit un mois et un jour, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. [Collectez une séquence de points de code](#) qui sont des caractères U+002D HYPHEN-MINUS (-) à partir de *la position* donnée *en entrée* . Si la séquence collectée ne contient pas exactement zéro ou deux caractères, échoue.
2. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée* . Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit le *mois* .



3. Si *le mois* n'est pas un nombre compris entre  $1 \leq \text{mois} \leq 12$ , alors échoue.
4. Soit *maxday* le [nombre de jours](#) dans le mois *mois* de toute année bissextile arbitraire (par exemple 4 ou 2000).
5. Si *la position* est au-delà de la fin de *l'entrée* ou si le caractère à *la position* n'est pas un caractère U+002D HYPHEN-MINUS, alors échoue. Sinon, déplacer *la position* vers l'avant d'un caractère.
6. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en entrée . Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit le *jour* .
7. Si *day* n'est pas un nombre compris entre  $1 \leq \text{day} \leq \text{maxday}$  , alors échoue.
8. Renvoie *le mois* et *le jour* .

#### 2.3.5.4 Heures

Une **heure** consiste en une heure spécifique sans information de fuseau horaire, composée d'une heure, d'une minute, d'une seconde et d'une fraction de seconde.

Une chaîne est une **chaîne de temps valide** représentant une heure *heure* , une minute *minute* et une seconde *seconde* si elle se compose des composants suivants dans l'ordre indiqué :

1. Deux [chiffres ASCII](#) , représentant *l' heure* , dans la plage  $0 \leq \text{heure} \leq 23$
2. Un caractère U+003A COLON (:) )
3. Deux [chiffres ASCII](#) , représentant *les minutes* , dans la plage  $0 \leq \text{minute} \leq 59$
4. Si *second* est différent de zéro, ou éventuellement si *second* est zéro :
  1. Un caractère U+003A COLON (:) )
  2. Deux [chiffres ASCII](#) , représentant la partie entière de *seconde* , dans la plage  $0 \leq s \leq 59$
  3. Si *second* n'est pas un entier, ou éventuellement si *second* est un entier :
    1. Un caractère U+002E POINT COMPLET (.) )
    2. Un, deux ou trois [chiffres ASCII](#) , représentant la partie fractionnaire de *la seconde*

*Le deuxième composant ne peut pas être 60 ou 61 ; les secondes intercalaires ne peuvent pas être représentées.*



Les règles d' **analyse d'une chaîne de temps** sont les suivantes. Cela renverra soit une heure, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Analyser un composant de temps](#) pour obtenir *l'heure* , *la minute* et *la seconde* . Si cela ne renvoie rien, échouez.
4. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
5. Soit *time* le temps avec heure *heure* , minute *minute* et seconde *seconde* .
6. *Heure* de retour .

Les règles d' **analyse d'un composant de temps** , étant donné une chaîne *d'entrée* et une *position* , sont les suivantes. Cela renverra soit une heure, une minute et une seconde, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée* . Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit *l' heure* .
2. Si *l'heure* n'est pas un nombre compris entre  $0 \leq \text{heure} \leq 23$ , alors échoue.
3. Si *la position* est au-delà de la fin de *la saisie* ou si le caractère à *la position* n'est pas un caractère U+003A COLON, alors échec. Sinon, déplacer *la position* vers l'avant d'un caractère.
4. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée* . Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit *la minute* .
5. Si *minute* n'est pas un nombre compris entre  $0 \leq \text{minute} \leq 59$ , alors échoue.
6. Soit *seconde égale* à 0.
7. Si *la position* n'est pas au-delà de la fin de *l'entrée* et que le caractère à *la position* est U+003A (:), alors :
  1. Avancez *la position* au caractère suivant dans *input* .
  2. Si *position* est au-delà de la fin de *input* , ou au dernier caractère de *input* , ou si les *deux* caractères suivants de *input* commençant à *position* ne sont pas tous deux [des chiffres ASCII](#) , alors échoue.

3. [Collectez une séquence de points de code](#) qui sont soit [des chiffres ASCII](#) , soit des caractères U+002E FULL STOP à partir de *la position* donnée *en entrée* . Si la séquence collectée comporte trois caractères, ou si elle comporte plus de trois caractères et que le troisième caractère n'est pas un caractère U+002E FULL STOP, ou si elle contient plus d'un caractère U+002E FULL STOP, l'échec échoue. Sinon, interprétez la séquence résultante comme un nombre en base dix (éventuellement avec une partie fractionnaire). Réglez *le deuxième* à ce nombre.
  4. Si *seconde* n'est pas un nombre compris entre  $0 \leq \textit{seconde} < 60$ , alors échoue.
8. Renvoie *l'heure* , *la minute* et *la seconde* .

### 2.3.5.5 Dates et heures locales

Une **date et une heure locales** consistent en une [date proleptique-grégorienne](#) spécifique, composée d'une année, d'un mois et d'un jour, et d'une heure, composée d'une heure, d'une minute, d'une seconde et d'une fraction de seconde, mais exprimée sans fuseau horaire. [\[GRÉGORIEN\]](#)

Une chaîne est une **chaîne de date et d'heure locale valide** représentant une date et une heure si elle se compose des composants suivants dans l'ordre indiqué :

1. Une [chaîne de date valide](#) représentant la date
2. Un caractère U+0054 LETTRE MAJUSCULE LATINE T (T) ou un caractère U+0020 ESPACE
3. Une [chaîne d'heure valide](#) représentant l'heure

Une chaîne est une **chaîne de date et d'heure locale normalisée valide** représentant une date et une heure si elle comprend les composants suivants dans l'ordre indiqué :

1. Une [chaîne de date valide](#) représentant la date
2. A U+0054 LETTRE MAJUSCULE LATINE T caractère (T)
3. Une [chaîne d'heure valide](#) représentant l'heure, exprimée sous la forme de la chaîne la plus courte possible pour l'heure donnée (par exemple, en omettant entièrement le composant secondes si l'heure donnée est à zéro seconde après la minute)

Les règles d' **analyse d'une chaîne de date et d'heure locale** sont les suivantes. Cela renverra soit une date et une heure, soit rien. Si à un moment donné

l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Analyser un composant de date](#) pour obtenir *l'année* , *le mois* et *le jour* . Si cela ne renvoie rien, échouez.
4. Si *la position* est au-delà de la fin de *la saisie* ou si le caractère à *la position* n'est ni un caractère U + 0054 LETTRE MAJUSCULE LATINE T (T) ni un caractère U + 0020 ESPACE, alors échec. Sinon, déplacer *la position* vers l'avant d'un caractère.
5. [Analyser un composant de temps](#) pour obtenir *l'heure* , *la minute* et *la seconde* . Si cela ne renvoie rien, échouez.
6. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
7. Soit *date* la date avec année *année* , mois *mois* et jour *jour* .
8. Soit *time* le temps avec heure *heure* , minute *minute* et seconde *seconde* .
9. *Date* et *heure* de retour .

#### 2.3.5.6 Fuseaux horaires

Un **décalage de fuseau horaire** consiste en un nombre signé d'heures et de minutes.

Une chaîne est une **chaîne de décalage de fuseau horaire valide** représentant un décalage de fuseau horaire si elle consiste en :

- A U+005A LETTRE MAJUSCULE LATINE Z caractère (Z), autorisé uniquement si le fuseau horaire est UTC
- Soit, les composants suivants, dans l'ordre indiqué :
  1. Soit un caractère U+002B PLUS SIGN (+) ou, si le décalage horaire n'est pas nul, un caractère U+002D TIRET MOINS (-), représentant le signe du décalage horaire
  2. Deux [chiffres ASCII](#) , représentant l'heure de la composante *heure* du décalage de fuseau horaire, dans la plage  $0 \leq \text{heure} \leq 23$
  3. Facultativement, un caractère U+003A COLON (:)

4. Deux [chiffres ASCII](#) , représentant la minute composant les *minutes* du décalage de fuseau horaire, dans la plage  $0 \leq \text{minute} \leq 59$

Ce format permet des décalages de fuseau horaire de -23:59 à +23:59. À l'heure actuelle, dans la pratique, la plage de décalages des fuseaux horaires réels est de -12h00 à +14h00, et la composante minutes des décalages des fuseaux horaires réels est toujours 00, 30 ou 45. Il n'y a aucune garantie que cela restera cependant pour toujours, puisque les fuseaux horaires sont utilisés comme ballons de football politiques et sont donc soumis à des décisions politiques très fantaisistes. Voir également les notes d'utilisation et les exemples dans la section [date et heure globales](#) ci-dessous pour plus de détails sur l'utilisation des décalages de fuseau horaire avec des heures historiques antérieures à la formation des fuseaux horaires formels.

Les règles d' **analyse d'une chaîne de décalage de fuseau horaire** sont les suivantes. Cela renverra soit un décalage de fuseau horaire, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Analysez un composant de décalage de fuseau horaire](#) pour obtenir *les heures* et *les minutes* du fuseau horaire . Si cela ne renvoie rien, échouez.
4. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
5. Renvoie le décalage de fuseau horaire qui est *le fuseau horaire heures* heures et les minutes *de fuseau horaire minutes* depuis UTC.

Les règles d' **analyse d'un composant de décalage de fuseau horaire** , étant donné une chaîne *d'entrée* et une *position* , sont les suivantes. Cela renverra soit les heures et les minutes du fuseau horaire, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Si le caractère à *la position* est un caractère U+005A LETTRE MAJUSCULE LATINE Z (Z), alors :
  1. Soit 0 pour *les heures* de fuseau horaire .
  2. Soit 0 pour *les minutes* du fuseau horaire .
  3. Avancez *la position* au caractère suivant dans *input* .

Sinon, si le caractère à *la position* est soit un SIGNE PLUS U+002B (+) soit un Trait d'union U+002D-MOINS (-), alors :

4. Si le caractère à *la position* est un SIGNE PLUS U+002B (+), laissez *le signe* être "positif". Sinon, c'est un trait d'union U+002D-MOINS (-); laissez *le signe* être "négatif".
5. Avancez *la position* au caractère suivant dans *input*.
6. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée*. Soit *s* la suite collectée.
7. Si *s* contient exactement deux caractères, alors :
  1. Interpréter *s* comme un entier de base dix. Soit ce nombre les *heures* du fuseau horaire.
  2. Si *la position* est au-delà de la fin de *la saisie* ou si le caractère à *la position* n'est pas un caractère U+003A COLON, alors échec. Sinon, déplacer *la position* vers l'avant d'un caractère.
  3. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée *en entrée*. Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Soit ce nombre les *minutes* du fuseau horaire.

Si *s* contient exactement quatre caractères, alors :

4. Interprétez les deux premiers caractères de *s* comme un entier de base dix. Soit ce nombre les *heures* du fuseau horaire.
5. Interprétez les deux derniers caractères de *s* comme un entier de base dix. Soit ce nombre les *minutes* du fuseau horaire.

Sinon, échouez.

8. Si les *heures* du fuseau horaire ne sont pas un nombre dans la plage  $0 \leq \text{heures du fuseau horaire} \leq 23$ , alors échec.
9. Si *le signe* est "négatif", alors annulez les *heures* du fuseau horaire.
10. Si les *minutes* du fuseau horaire ne sont pas un nombre dans la plage  $0 \leq \text{minutes du fuseau horaire} \leq 59$ , alors échec.
11. Si *le signe* est "négatif", alors annulez les *minutes* du fuseau horaire.

Sinon, échouez.

2. Renvoie les *heures* et les *minutes* du fuseau horaire.

### 2.3.5.7 Dates et heures globales

Une **date et une heure globales** consistent en une [date proleptique-grégorienne](#) spécifique, composée d'une année, d'un mois et d'un jour, et d'une heure, composée d'une heure, d'une minute, d'une seconde et d'une fraction de seconde, exprimée par un décalage de fuseau horaire, composé d'un nombre signé d'heures et de minutes. [\[GRÉGORIEN\]](#)

Une chaîne est une **chaîne de date et d'heure globale valide** représentant une date, une heure et un décalage de fuseau horaire si elle comprend les composants suivants dans l'ordre indiqué :

1. Une [chaîne de date valide](#) représentant la date
2. Un caractère U+0054 LETTRE MAJUSCULE LATINE T (T) ou un caractère U+0020 ESPACE
3. Une [chaîne d'heure valide](#) représentant l'heure
4. Une [chaîne de décalage de fuseau horaire valide](#) représentant le décalage de fuseau horaire

Les heures des dates antérieures à la formation de l'UTC au milieu du XXe siècle doivent être exprimées et interprétées en termes d'UT1 (heure solaire terrestre contemporaine à la longitude 0°), et non d'UTC (l'approximation d'UT1 qui s'égrène en secondes SI). L'heure avant la formation des fuseaux horaires doit être exprimée et interprétée comme des heures UT1 avec des fuseaux horaires explicites qui se rapprochent de la différence contemporaine entre l'heure locale appropriée et l'heure observée à l'emplacement de Greenwich, Londres.

Voici quelques exemples de dates écrites sous forme [de chaînes de date et d'heure globales valides](#).

```
" 0037-12-13 00:00Z"
```

Minuit dans les zones utilisant l'heure de Londres le jour de l'anniversaire de Néron (l'empereur romain). Voir ci-dessous pour plus de détails sur la date à laquelle cela correspond réellement.

```
" 1979-10-14T12:00:00.001-04:00"
```

Une milliseconde après midi le 14 octobre 1979, dans le fuseau horaire en vigueur sur la côte est des États-Unis à l'heure d'été.

```
" 8592-01-01T02:09+02:09"
```

Minuit UTC le 1er janvier 8592. Le fuseau horaire associé à cette heure a deux heures et neuf minutes d'avance sur UTC, qui n'est actuellement pas un fuseau horaire réel, mais qui est néanmoins autorisé.

Plusieurs choses sont remarquables à propos de ces dates :

- Les années à moins de quatre chiffres doivent être complétées par des zéros. La date "37-12-13" ne serait pas une date valide.

- Si le " T " est remplacé par un espace, il doit s'agir d'un espace unique. La chaîne " 2001-12-21 12:00Z " (avec deux espaces entre les composants) ne serait pas analysée correctement.
- Pour identifier sans ambiguïté un moment dans le temps avant l'introduction du calendrier grégorien (dans la mesure où les moments dans le temps avant la formation de l'UTC peuvent être identifiés sans ambiguïté), la date doit d'abord être convertie au calendrier grégorien à partir du calendrier en usage à l'heure (par exemple du calendrier julien). La date de naissance de Néron est le 15 décembre 37, dans le calendrier julien, qui est le 13 décembre 37 dans le [calendrier grégorien proleptique](#) .
- Les composants de décalage horaire et de fuseau horaire ne sont pas facultatifs.
- Les dates antérieures à l'année un ne peuvent pas être représentées par une date et une heure dans cette version de HTML.
- Les heures d'événements spécifiques dans les temps anciens sont, au mieux, des approximations, car le temps n'était pas bien coordonné ou mesuré jusqu'à des décennies relativement récentes.
- Les décalages de fuseau horaire diffèrent en fonction de l'heure d'été.

Les règles d' **analyse d'une chaîne de date et d'heure globales** sont les suivantes. Cela renverra soit une heure en UTC, avec les informations de décalage de fuseau horaire associées à des fins d'aller-retour ou d'affichage, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Analyser un composant de date](#) pour obtenir *l'année* , *le mois* et *le jour* . Si cela ne renvoie rien, échouez.
4. Si *la position* est au-delà de la fin de *la saisie* ou si le caractère à *la position* n'est ni un caractère U + 0054 LETTRE MAJUSCULE LATINE T (T) ni un caractère U + 0020 ESPACE, alors échec. Sinon, déplacer *la position* vers l'avant d'un caractère.
5. [Analyser un composant de temps](#) pour obtenir *l'heure* , *la minute* et *la seconde* . Si cela ne renvoie rien, échouez.
6. Si *la position* est au-delà de la fin de *input* , alors échoue.
7. [Analysez un composant de décalage de fuseau horaire](#) pour obtenir *les heures* et *les minutes* du fuseau horaire . Si cela ne renvoie rien, échouez.
8. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.



9. Soit *time* le moment dans le temps à année *année* , mois *mois* , jour *jour* , heures *heure* , minute *minute* , seconde *seconde* , en soustrayant <sup>les heures</sup> *du fuseau horaire* heures et les <sup>minutes</sup> *du fuseau horaire* minutes. Ce moment dans le temps est un moment dans le fuseau horaire UTC.
10. Soit *le fuseau horaire* être <sup>les heures</sup> *heures* *du fuseau horaire* et les <sup>minutes</sup> *du fuseau horaire* minutes depuis UTC.
11. *Heure de retour et fuseau horaire* .

### 2.3.5.8 Semaines

Une **semaine** se compose d'un numéro de semaine-année et d'un numéro de semaine représentant une période de sept jours commençant un lundi. Chaque année-semaine dans ce système de calendrier a 52 ou 53 périodes de sept jours, comme défini ci-dessous. La période de sept jours commençant à la date grégorienne du lundi 29 décembre 1969 (1969-12-29) est définie comme la semaine numéro 1 de l'année-semaine 1970. Les semaines consécutives sont numérotées séquentiellement. La semaine précédant la semaine numéro 1 dans une semaine-année est la dernière semaine de la semaine-année précédente, et vice versa. [\[GRÉGORIEN\]](#)

Une année-semaine avec un nombre *année* a 53 semaines si elle correspond soit à une année *année* du [calendrier grégorien proleptique](#) qui a un jeudi comme premier jour (1er janvier), soit à une année *année* du calendrier [grégorien proleptique](#) qui a un mercredi comme son premier jour (1er janvier) et où *l'année* est un nombre divisible par 400, ou un nombre divisible par 4 mais pas par 100. Toutes les autres semaines-années ont 52 semaines.

Le **numéro de semaine du dernier jour** d'une année-semaine de 53 semaines est 53 ; le numéro de semaine du dernier jour d'une semaine-année de 52 semaines est 52.

*Le numéro de semaine-année d'un jour particulier peut être différent du numéro de l'année qui contient ce jour dans le [calendrier grégorien proleptique](#) . La première semaine d'une semaine-année y est la semaine qui contient le premier jeudi de l'année grégorienne y .*  
*À des fins modernes, une [semaine](#) telle que définie ici équivaut à des semaines ISO telles que définies dans la norme ISO 8601. [\[ISO8601\]](#)*

Une chaîne est une **chaîne de semaine valide** représentant une année-semaine et une semaine- *semaine* si elle comprend les composants suivants dans l'ordre indiqué :

1. Quatre [chiffres ASCII](#) ou plus , représentant *l' année* , où *année* > 0
2. Un caractère U+002D TIRET-MOINS (-)



3. A U+0057 LETTRE MAJUSCULE LATINE W caractère (W)
4. Deux chiffres ASCII , représentant la semaine *week* , dans la plage  $1 \leq week \leq maxweek$  , où *maxweek* est le numéro de semaine du dernier jour de la semaine- *année*

Les règles pour **analyser une chaîne de semaine** sont les suivantes. Cela renverra soit un numéro de semaine-année et un numéro de semaine, soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. Collectez une séquence de points de code qui sont des chiffres ASCII à partir de *la position* donnée *en entrée* . Si la séquence collectée ne comporte pas au moins quatre caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit l' *année* .
4. Si l'*année* n'est pas un nombre supérieur à zéro, échouez.
5. Si *la position* est au-delà de la fin de l'*entrée* ou si le caractère à *la position* n'est pas un caractère U+002D HYPHEN-MINUS, alors échoue. Sinon, déplacer *la position* vers l'avant d'un caractère.
6. Si *la position* est au-delà de la fin de *la saisie* ou si le caractère à *la position* n'est pas un caractère U + 0057 LETTRE MAJUSCULE LATINE W (W), alors échoue. Sinon, déplacer *la position* vers l'avant d'un caractère.
7. Collectez une séquence de points de code qui sont des chiffres ASCII à partir de *la position* donnée *en entrée* . Si la séquence collectée ne contient pas exactement deux caractères, échoue. Sinon, interprétez la séquence résultante comme un entier de base dix. Que ce nombre soit la *semaine* .
8. Soit *maxweek* le numéro de la semaine du dernier jour de l'année *year* .
9. Si *week* n'est pas un nombre compris entre  $1 \leq week \leq maxweek$  , alors échoue.
10. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
11. Renvoie le numéro de semaine-année *year* et le numéro de semaine *week* .

### 2.3.5.9 Durées

Une **durée** est constituée d'un nombre de secondes.

*Étant donné que les mois et les secondes ne sont pas comparables (un mois n'est pas un nombre précis de secondes, mais plutôt une période dont la durée exacte dépend du jour précis à partir duquel elle est mesurée), une durée telle que définie dans cette spécification ne peut pas inclure des mois (ou des années, qui équivalent à douze mois). Seules les durées décrivant un nombre spécifique de secondes peuvent être décrites.*

Une chaîne est une **chaîne de durée valide** représentant une durée  $t$  si elle se compose de l'un des éléments suivants :

- Un caractère littéral U+0050 LETTRE MAJUSCULE LATINE P suivi d'un ou plusieurs des sous-composants suivants, dans l'ordre indiqué, où le nombre de jours, d'heures, de minutes et de secondes correspond au même nombre de secondes que dans  $t$  :
  1. Un ou plusieurs chiffres ASCII suivis d'un caractère U+0044 LETTRE MAJUSCULE LATINE D, représentant un nombre de jours.
  2. Un caractère U+0054 LETTRE MAJUSCULE LATINE T suivi d'un ou plusieurs des sous-composants suivants, dans l'ordre indiqué :
    1. Un ou plusieurs chiffres ASCII suivis d'un caractère U+0048 LETTRE MAJUSCULE LATINE H, représentant un nombre d'heures.
    2. Un ou plusieurs chiffres ASCII suivis d'un caractère U+004D LETTRE MAJUSCULE LATINE M, représentant un nombre de minutes.
  3. Les composants suivants :
    1. Un ou plusieurs chiffres ASCII, représentant un nombre de secondes.
    2. Facultativement, un caractère U+002E FULL STOP (.) suivi d'un, deux ou trois chiffres ASCII, représentant une fraction de seconde.
    3. A U+0053 LETTRE MAJUSCULE LATINE S caractère.

*Ceci, comme avec un certain nombre d'autres microsyntaxes liées à la date et à l'heure définies dans cette spécification, est basé sur l'un des formats définis dans l'ISO 8601. [\[ISO8601\]](#)*

- Un ou plusieurs composants de temps de durée, chacun avec une échelle de composants de temps de durée différente, dans n'importe quel ordre ; la somme des secondes représentées étant égale au nombre de secondes dans  $t$ .

Un **composant de durée** est une chaîne constituée des composants suivants :

1. Zéro ou plusieurs [espaces blancs ASCII](#) .
2. Un ou plusieurs [chiffres ASCII](#) , représentant un certain nombre d'unités de temps, mis à l'échelle par l' [échelle de composante de temps de durée](#) spécifiée (voir ci-dessous) pour représenter un nombre de secondes.
3. Si l' [échelle de la composante temporelle de durée](#) spécifiée est 1 (c'est-à-dire que les unités sont des secondes), alors, éventuellement, un caractère U+002E FULL STOP (.) suivi d'un, deux ou trois [chiffres ASCII](#) , représentant une fraction de seconde.
4. Zéro ou plusieurs [espaces blancs ASCII](#) .
5. L'un des caractères suivants, représentant l' **échelle de la composante temporelle de la durée** de l'unité de temps utilisée dans la partie numérique de la [composante temporelle de la durée](#) :

**U+0057 LETTRE MAJUSCULE LATINE W caractère**

**U+0077 LETTRE MINUSCULE LATINE W caractère**

Semaines. L'échelle est 604800.

**U+0044 LETTRE MAJUSCULE LATINE D caractère**

**U+0064 LETTRE MINUSCULE LATINE D caractère**

Jours. L'échelle est 86400.

**U+0048 LETTRE MAJUSCULE LATINE H caractère**

**U+0068 LETTRE MINUSCULE LATINE H caractère**

Heures. L'échelle est de 3600.

**U+004D LETTRE MAJUSCULE LATINE M caractère**

**U+006D LETTRE MINUSCULE LATINE M caractère**

Minutes. L'échelle est de 60.

**U+0053 LETTRE MAJUSCULE LATINE S caractère**

**U+0073 LETTRE MINUSCULE LATINE S caractère**

Secondes. L'échelle est 1.

6. Zéro ou plusieurs [espaces blancs ASCII](#) .

*Ceci n'est basé sur aucun des formats de la norme ISO 8601. Il est destiné à être une alternative plus lisible par l'homme au format de durée ISO 8601.*

Les règles d' **analyse d'une chaîne de durée** sont les suivantes. Cela renverra soit une [durée](#) , soit rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.

2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. Laissez *les mois* , *les secondes* et *le nombre de composants* à zéro.
4. Soit *M-disambiguator* être *minutes* .

*L'autre valeur de cet indicateur est mois . Il est utilisé pour lever l'ambiguïté de l'unité "M" dans les durées ISO8601, qui utilisent la même unité pour les mois et les minutes. Les mois ne sont pas autorisés, mais sont analysés pour une compatibilité future et pour éviter une mauvaise interprétation des durées ISO8601 qui seraient valides dans d'autres contextes.*

5. [Ignorer les espaces blancs ASCII](#) dans *la position d'entrée* donnée .
6. Si *la position* dépasse la fin de *input* , alors échoue.
7. Si le caractère en *entrée* pointé par *position* est un caractère U+0050 LETTRE MAJUSCULE LATINE P, alors avancez *la position* au caractère suivant, réglez *M-disambiguator* sur *mois* , et [ignorez les espaces blancs ASCII](#) dans *la position* donnée en *entrée* .
8. Alors que c'est vrai :
  1. Soit *les unités* indéfinies. L'une des valeurs suivantes lui sera attribuée : *ans* , *mois* , *semaines* , *jours* , *heures* , *minutes* et *secondes* .
  2. Laissez *le caractère suivant* être indéfini. Il est utilisé pour traiter les caractères de l' *entrée* .
  3. Si *position* dépasse la fin de *input* , alors break.
  4. Si le caractère en *entrée* pointé par *position* est un caractère U+0054 LETTRE MAJUSCULE LATINE T, alors avancez *position* au caractère suivant, réglez *M-disambiguator* sur *minutes* , [ignorez les espaces blancs ASCII](#) dans *la position* donnée en *entrée* et [continuez](#) .
  5. Définissez le *caractère suivant* sur le caractère en *entrée* pointé par *position* .
  6. Si *le caractère suivant* est un caractère U+002E FULL STOP (.), alors laissez *N* égal à zéro. (N'avancez pas *de position* . Cela est pris en compte ci-dessous.)

Sinon, si *le caractère suivant* est un [chiffre ASCII](#) , [collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en *entrée* , interprétez la séquence résultante comme un entier de base dix et laissez *N* être ce nombre.

Sinon, *le caractère suivant* ne fait pas partie d'un nombre ; échouer.
  7. Si *la position* dépasse la fin de *input* , alors échoue.

8. Définissez le *caractère suivant* sur le caractère en *entrée* pointé par *position* , et cette fois avancez *la position* au caractère suivant. (Si le *caractère suivant* était un caractère U+002E FULL STOP (.) auparavant, ce sera toujours ce caractère cette fois.)
9. Si le *caractère suivant* est U+002E (.) , alors :
  1. Collectez une séquence de points de code qui sont des chiffres ASCII à partir de *la position* donnée en *entrée* . Soit *s* la suite résultante.
  2. Si *s* est la chaîne vide, échoue.
  3. Soit *longueur* le nombre de caractères dans *s* .
  4. Soit *fraction* le résultat de l'interprétation de *s* comme un entier de base dix, puis de la division de ce nombre par  $10^{length}$  .
  5. Incrémenter *N* par *fraction* .
  6. Ignorer les espaces blancs ASCII dans *la position d'entrée* donnée .
  7. Si *la position* dépasse la fin de *input* , alors échoue.
  8. Définissez le *caractère suivant* sur le caractère d' *entrée* pointé par *position* et avancez *la position* jusqu'au caractère suivant.
  9. Si le *caractère suivant* n'est ni un caractère U+0053 LETTRE MAJUSCULE LATINE S ni un caractère U+0073 LETTRE MINUSCULE LATINE S, alors échec.
  10. Réglez *les unités* sur *les secondes* .

Sinon:

11. Si le *caractère suivant* est un espace blanc ASCII , ignorez l'espace blanc ASCII dans *la position* donnée en *entrée* , définissez le *caractère suivant* sur le caractère d' *entrée* pointé par *position* et avancez *la position* au caractère suivant.
12. Si le *caractère suivant* est un caractère U+0059 LETTRE MAJUSCULE LATINE Y ou un caractère U+0079 LETTRE MINUSCULE LATINE Y, réglez *les unités* sur *les années* et réglez le *désambiguïsateur M* sur *les mois* .

Si le *caractère suivant* est un caractère U+004D LETTRE MAJUSCULE LATINE M ou un caractère U+006D LETTRE MINUSCULE LATINE M et que le *désambiguïsateur M* est *mois* , alors réglez *les unités* sur *mois* .

Si *le caractère suivant* est un caractère U+0057 LETTRE MAJUSCULE LATINE W ou un caractère U+0077 LETTRE MINUSCULE LATINE W, réglez *les unités* sur *les semaines* et réglez *le désambiguïsateur M* sur *les minutes* .

Si *le caractère suivant* est un caractère U+0044 LETTRE MAJUSCULE LATINE D ou un caractère U+0064 LETTRE MINUSCULE LATINE D, réglez *les unités* sur *les jours* et réglez *le désambiguïsateur M* sur *les minutes* .

Si *le caractère suivant* est un caractère U+0048 LETTRE MAJUSCULE LATINE H ou un caractère U+0068 LETTRE MINUSCULE LATINE H, réglez *les unités* sur *les heures* et réglez *le désambiguïsateur M* sur *les minutes* .

Si *le caractère suivant* est un caractère U+004D LETTRE MAJUSCULE LATINE M ou un caractère U+006D LETTRE MINUSCULE LATINE M et que *le désambiguïsateur M* est *minutes* , alors réglez *les unités* sur *minutes* .

Si *le caractère suivant* est un caractère U+0053 LETTRE MAJUSCULE LATINE S ou un caractère U+0073 LETTRE MINUSCULE LATINE S, réglez *les unités* sur *les secondes* et réglez *le désambiguïsateur M* sur *les minutes* .

Sinon, si *le caractère suivant* n'est aucun des caractères ci-dessus, échoue.

10. Incrémenter *le nombre de composants* .

11. Soit *le multiplicateur* égal à 1.

12. Si *les unités* sont *des années* , multipliez *le multiplicateur* par 12 et définissez *les unités* sur *les mois* .

13. Si *les unités* sont *des mois* , ajoutez le produit de *N* et *le multiplicateur* aux *mois* .

Sinon:

1. Si *les unités* sont *des semaines* , multipliez *le multiplicateur* par 7 et réglez *les unités* sur *les jours* .
2. Si *les unités* sont *des jours* , multipliez *le multiplicateur* par 24 et réglez *les unités* sur *les heures* .
3. Si *les unités* sont *des heures* , multipliez *le multiplicateur* par 60 et réglez *les unités* sur *les minutes* .
4. Si *les unités* sont *des minutes* , multipliez *le multiplicateur* par 60 et réglez *les unités* sur *les secondes* .

5. Forcément, *les unités* sont maintenant *des secondes* . Ajouter le produit de *N* et *le multiplicateur* aux *secondes* .

14. [Ignorer les espaces blancs ASCII](#) dans *la position d'entrée* donnée .

9. Si *le nombre de composants* est zéro, échec.

10. Si *mois* n'est pas égal à zéro, échec.

11. Renvoie la [durée](#) composée de *secondes* secondes.

### 2.3.5.10 Moments plus vagues dans le temps

Une chaîne est une **chaîne de date valide avec une heure facultative** si elle est également l'une des suivantes :

- Une [chaîne de date valide](#)
- Une [chaîne de date et d'heure globale valide](#)

---

Les règles d' **analyse d'une chaîne de date ou d'heure** sont les suivantes. L'algorithme renverra soit une [date](#) , une [heure](#) , une [date et une heure globales](#) , ou rien. Si à un moment donné l'algorithme dit qu'il "échoue", cela signifie qu'il est abandonné à ce moment-là et ne renvoie rien.

1. Soit *input* la chaîne analysée.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. Réglez *la position de départ* sur la même position que *la position* .
4. Définissez les indicateurs *de date* et *d'heure* sur true.
5. [Analyser un composant de date](#) pour obtenir *l'année* , *le mois* et *le jour* . Si cela échoue, définissez l'indicateur *de date actuelle* sur faux.
6. Si *la date présente* est vraie et que *la position* n'est pas au-delà de la fin de *l'entrée* et que le caractère à *la position* est soit un caractère U + 0054 LETTRE MAJUSCULE LATINE T (T) ou un caractère U + 0020 ESPACE, alors avancez *la position* au caractère suivant en *entrée* .



Sinon, si *la date présente* est vraie et que *la position* est au-delà de la fin de *l'entrée* ou que le caractère à *la position* n'est ni un caractère U + 0054 LETTRE MAJUSCULE LATINE T (T) ni un caractère U + 0020 ESPACE, alors définissez *le temps présent* sur faux .

Sinon, si *la date présente* est fausse, remettez *la position* à la même position que *la position de départ* .

7. Si l'indicateur *de temps présent* est vrai, [analysez un composant de temps](#) pour obtenir *heure* , *minute* et *seconde* . Si cela ne renvoie rien, échouez.
8. Si les indicateurs *de date* et *d'heure présents* sont tous les deux vrais, mais que *la position* est au-delà de la fin de *input* , alors échoue.
9. Si les indicateurs *de date actuelle* et *d'heure actuelle* sont tous les deux vrais, [analysez un composant de décalage de fuseau horaire](#) pour obtenir <sup>les</sup> heures et <sup>les minutes</sup> *de fuseau horaire* . Si cela ne renvoie rien, échouez.
10. Si *la position* n'est pas au-delà de la fin de *input* , alors échoue.
11. Si l'indicateur *de date actuelle* est vrai et que l'indicateur *d'heure actuelle* est faux, alors *date* sera la date avec l'année *année* , le mois *mois* et le jour *jour* et *la date* de retour .

Sinon, si l'indicateur *de temps présent* est vrai et que l'indicateur *de date présente* est faux, alors laissez *time* être l'heure avec heure *heure* , minute *minute* et seconde *seconde* , et *temps* de retour .

Sinon, laissez *time* être le moment dans le temps à l'année *year* , month *month* , day *day* , hours *hour* , minute *minute* , second *second* , en soustrayant *le fuseau horaire* <sup>heures</sup> heures et *le fuseau horaire* <sup>minutes</sup> minutes, ce moment dans le temps étant un moment dans le fuseau horaire UTC ; laissez *le fuseau horaire* être <sup>les heures heures</sup> *du fuseau horaire* et les <sup>minutes</sup> *du fuseau horaire* depuis UTC ; et retour de *l'heure* et *du fuseau horaire* .

### 2.3.6 Couleurs

Une **couleur simple** se compose de trois nombres de 8 bits dans la plage de 0 à 255 inclus, représentant respectivement les composants rouge, vert et bleu de la couleur, dans l'espace colorimétrique '[srgb](#)' .

Une chaîne est une **couleur simple valide** si elle contient exactement sept caractères, et le premier caractère est un caractère NUMÉRO U+0023 (#), et les six caractères restants sont tous des [chiffres hexadécimaux ASCII](#) , les deux premiers chiffres représentant le rouge composante, les deux chiffres du milieu représentant la



composante verte et les deux derniers chiffres représentant la composante bleue, en hexadécimal.

Une chaîne est une **couleur simple minuscule valide** si elle est une [couleur simple valide](#) et n'utilise aucun caractère dans la plage U+0041 LETTRE MAJUSCULE LATINE A à U+0046 LETTRE MAJUSCULE LATINE F.

Les **règles d'analyse des valeurs de couleur simples** sont données dans l'algorithme suivant. Lorsqu'elles sont invoquées, les étapes doivent être suivies dans l'ordre indiqué, en abandonnant à la première étape qui renvoie une valeur. Cet algorithme renverra soit une [simple couleur](#) , soit une erreur.

1. Soit *input* la chaîne analysée.
2. Si *l'entrée* ne contient pas exactement sept caractères, renvoie une erreur.
3. Si le premier caractère *saisi* n'est pas un caractère NUMÉRO U+0023 (#), renvoie une erreur.
4. Si les six derniers caractères d' *entrée* ne sont pas tous [des chiffres hexadécimaux ASCII](#) , renvoie une erreur.
5. Soit *le résultat* une [couleur simple](#) .
6. Interprétez les deuxième et troisième caractères comme un nombre hexadécimal et laissez le résultat être la composante rouge de *result* .
7. Interprétez les quatrième et cinquième caractères comme un nombre hexadécimal et laissez le résultat être la composante verte de *result* .
8. Interprétez les sixième et septième caractères comme un nombre hexadécimal et laissez le résultat être la composante bleue de *result* .
9. Retourner *le résultat* .

Les **règles de sérialisation des valeurs de couleur simples** pour une [couleur simple](#) sont données dans l'algorithme suivant :

1. Soit *result* une chaîne composée d'un seul caractère U+0023 NUMBER SIGN (#).
  2. Convertissez les composants rouge, vert et bleu tour à tour en nombres hexadécimaux à deux chiffres en utilisant les [chiffres hexadécimaux inférieurs ASCII](#) , en ajoutant des zéros si nécessaire, et ajoutez ces nombres au *résultat* , dans l'ordre rouge, vert, bleu.
  3. Renvoie *le résultat* , qui sera une [couleur simple en minuscule valide](#) .
-

Certains attributs hérités obsolètes analysent les couleurs de manière plus compliquée, en utilisant les **règles d'analyse d'une valeur de couleur héritée**, qui sont données dans l'algorithme suivant. Lorsqu'elles sont invoquées, les étapes doivent être suivies dans l'ordre indiqué, en abandonnant à la première étape qui renvoie une valeur. Cet algorithme renverra soit une [simple couleur](#), soit une erreur.

1. Soit *input* la chaîne analysée.
2. Si *l'entrée* est la chaîne vide, renvoie une erreur.
3. [Supprime les espaces blancs ASCII de début et de fin](#) de *l'entrée*.
4. Si *l'entrée* est une correspondance [ASCII non sensible à la casse](#) pour la chaîne "transparent", renvoie une erreur.
5. Si *l'entrée* est une correspondance [ASCII non sensible à la casse](#) pour l'une des [couleurs nommées](#), renvoie la [couleur simple](#) correspondant à ce mot-clé. [\[CSSCOLOR\]](#)

[Les couleurs système CSS2](#) ne sont pas reconnues.

6. Si [la longueur du point de code](#) de *l'entrée* est de quatre, et que le premier caractère de *l'entrée* est U+0023 (#), et que les trois derniers caractères de *l'entrée* sont tous [des chiffres hexadécimaux ASCII](#), alors :
  1. Soit *le résultat* une [couleur simple](#).
  2. Interprétez le deuxième caractère d' *entrée* comme un chiffre hexadécimal ; soit la composante rouge du *résultat* soit le nombre résultant multiplié par 17.
  3. Interprétez le troisième caractère d' *entrée* comme un chiffre hexadécimal ; soit la composante verte du *résultat* soit le nombre résultant multiplié par 17.
  4. Interprétez le quatrième caractère d' *entrée* comme un chiffre hexadécimal ; soit la composante bleue du *résultat* soit le nombre résultant multiplié par 17.
5. Retourner *le résultat*.
7. Remplacez tous [les points de code](#) supérieurs à U+FFFF en *entrée* (c'est-à-dire tous les caractères qui ne sont pas dans le plan multilingue de base) par la chaîne de deux caractères " " 00.
8. Si [la longueur du point de code](#) de *input* est supérieure à 128, tronque *input*, ne laissant que les 128 premiers caractères.
9. Si le premier caractère *saisi* est un caractère NUMÉRO U+0023 (#), supprimez-le.

10. Remplacez tout caractère en *entrée* qui n'est pas un [chiffre hexadécimal ASCII](#) par le caractère U+0030 DIGIT ZERO (0).
  11. Alors que [la longueur du point de code](#) de *input* est zéro ou n'est pas un multiple de trois, ajoutez un caractère U+0030 DIGIT ZERO (0) à *input*.
  12. Divisez *l'entrée* en trois chaînes de [longueur de point de code](#) égale pour obtenir trois composants. Soit *length* la [longueur en points de code](#) de tous ces composants (un tiers de la [longueur en points de code](#) de *input*).
  13. Si *la longueur* est supérieure à 8, supprimez la *longueur* de tête -8 caractères dans chaque composant et laissez *la longueur* être 8.
  14. Lorsque *la longueur* est supérieure à deux et que le premier caractère de chaque composant est un caractère U+0030 DIGIT ZERO (0), supprimez ce caractère et réduisez *la longueur* de un.
  15. Si *la longueur* est *toujours* supérieure à deux, tronquez chaque composant, en ne laissant que les deux premiers caractères dans chacun.
  16. Soit *le résultat* une [couleur simple](#).
  17. Interprétez le premier composant comme un nombre hexadécimal ; laissez la composante rouge du *résultat* être le nombre résultant.
  18. Interprétez le deuxième composant comme un nombre hexadécimal ; laissez la composante verte du *résultat* être le nombre résultant.
  19. Interprétez le troisième composant comme un nombre hexadécimal ; laissez la composante bleue du *résultat* être le nombre résultant.
  20. Retourner *le résultat*.
- 

Le [contexte graphique 2D](#) a une syntaxe de couleur distincte qui gère également l'opacité.

### 2.3.7 Jetons séparés par des espaces

Un **ensemble de jetons séparés par des espaces** est une chaîne contenant zéro ou plusieurs mots (appelés jetons) séparés par un ou plusieurs [espaces blancs ASCII](#), où les mots consistent en une chaîne d'un ou plusieurs caractères, dont aucun n'est [un espace blanc ASCII](#).

Une chaîne contenant un [ensemble de jetons séparés par des espaces](#) peut avoir [des espaces blancs ASCII](#) de début ou de fin .

Un **ensemble non ordonné de jetons uniques séparés par des espaces** est un [ensemble de jetons séparés par des espaces](#) où aucun des jetons n'est dupliqué.

Un **ensemble ordonné de jetons uniques séparés par des espaces** est un [ensemble de jetons séparés par des espaces](#) où aucun des jetons n'est dupliqué mais où l'ordre des jetons est significatif.

[Les ensembles de jetons séparés par des espaces](#) ont parfois un ensemble défini de valeurs autorisées. Lorsqu'un ensemble de valeurs autorisées est défini, les jetons doivent tous provenir de cette liste de valeurs autorisées ; les autres valeurs ne sont pas conformes. Si aucun ensemble de valeurs autorisées n'est fourni, toutes les valeurs sont conformes.

*La manière dont les jetons d'un [ensemble de jetons séparés par des espaces](#) doivent être comparés (par exemple, en respectant ou non la casse) est définie pour chaque ensemble.*

### 2.3.8 Jetons séparés par des virgules

Un **ensemble de jetons séparés par des virgules** est une chaîne contenant zéro ou plusieurs jetons, chacun séparé du suivant par un seul caractère virgule U + 002C (,), où les jetons consistent en une chaîne de zéro ou plusieurs caractères, ne commençant ni ne se terminant par [ASCII whitespace](#) , ne contenant aucun caractère virgule U+002C (,) et éventuellement entouré d' [un espace blanc ASCII](#) .

Par exemple, la chaîne " a , b , , d d " se compose de quatre jetons : "a", "b", la chaîne vide et "d d". Les espaces blancs de début et de fin autour de chaque jeton ne comptent pas comme faisant partie du jeton, et la chaîne vide peut être un jeton.

[Les ensembles de jetons séparés par des virgules](#) ont parfois des restrictions supplémentaires sur ce qui constitue un jeton valide. Lorsque de telles restrictions sont définies, les jetons doivent tous s'adapter à ces restrictions ; les autres valeurs ne sont pas conformes. Si aucune restriction de ce type n'est spécifiée, toutes les valeurs sont conformes.

### 2.3.9 Références

Une **référence de nom de hachage valide** à un élément de type *type* est une chaîne composée d'un caractère NUMÉRO U+0023 (#) suivi d'une chaîne qui

correspond exactement à la valeur de l' `name` attribut d'un élément de type *type* dans le même [arbre](#) .

Les **règles d'analyse d'une référence de nom de hachage** à un élément de type *type* , étant donné une *portée* de nœud de contexte , sont les suivantes :

1. Si la chaîne en cours d'analyse ne contient pas de caractère NUMÉRO U+0023, ou si le premier caractère de ce type dans la chaîne est le dernier caractère de la chaîne, renvoie null.
2. Soit *s* la chaîne du caractère immédiatement après le premier caractère U+0023 NUMBER SIGN dans la chaîne analysée jusqu'à la fin de cette chaîne.
3. Renvoie le premier élément de type *type* dans l' [arborescence](#) de la *portée* , dans l'[ordre de l'arborescence](#) , qui a un attribut ou dont la valeur est *s* , ou null s'il n'y a pas un tel élément. `idname`

*Bien que `id` les attributs soient pris en compte lors de l'analyse, ils ne sont pas utilisés pour déterminer si une valeur est une [référence de nom de hachage valide](#) . Autrement dit, une référence de nom de hachage qui fait référence à un élément basé sur `id` est une erreur de conformité (à moins que cet élément ait également un `name` attribut avec la même valeur).*

### 2.3.10 Requêtes média

Une chaîne est une **liste de requêtes multimédia valide** si elle correspond à la `<media-query-list>` production de *Media Queries* . [\[QM\]](#)

Une chaîne **correspond à l'environnement** de l'utilisateur s'il s'agit d'une chaîne vide, d'une chaîne composée uniquement d' [espaces ASCII](#) ou d'une liste de requêtes multimédias qui correspond à l'environnement de l'utilisateur selon les définitions données dans *Media Queries* . [\[QM\]](#)

### 2.3.11 Valeurs internes uniques

Une **valeur interne unique** est une valeur qui est sérialisable, comparable par valeur et jamais exposée au script.

Pour créer une **nouvelle valeur interne unique** , renvoyez une [valeur interne unique](#) qui n'a jamais été renvoyée auparavant par cet algorithme.

## 2.4 URL

### 2.4.1 Terminologie

Une chaîne est une **URL non vide valide** s'il s'agit d'une [chaîne d'URL valide](#) mais qu'il ne s'agit pas de la chaîne vide.

Une chaîne est une **URL valide potentiellement entourée d'espaces** si, après [avoir supprimé les espaces blancs ASCII de début et](#) de fin, il s'agit d'une [chaîne d'URL valide](#) .

Une chaîne est une **URL non vide valide potentiellement entourée d'espaces** si, après [avoir supprimé les espaces blancs ASCII de début et](#) de fin, il s'agit d'une [URL non vide valide](#) .

Cette spécification définit l'URL [about:legacy-compat](#) comme une URL réservée, bien qu'insoluble [about:](#), à utiliser dans les [DOCTYPE](#) des [documents HTML](#) lorsque cela est nécessaire pour la compatibilité avec les outils XML. [\[À PROPOS DE\]](#)

Cette spécification définit l'URL [about:html-kind](#) comme une URL réservée, bien qu'insoluble [about:](#), qui est utilisée comme identifiant pour les types de pistes multimédias. [\[À PROPOS DE\]](#)

Cette spécification définit l'URL [about:srcdoc](#) comme une [about:](#) URL réservée, bien qu'insoluble, qui est utilisée comme [URL](#) des [iframe srcdoc documents](#) . [\[À PROPOS DE\]](#)

L' **URL de base de secours** d'un [document](#) [Document](#) objet est l' [enregistrement d'URL](#) obtenu en exécutant ces étapes :

1. Si *document* est [un iframe srcdoc document](#) , renvoie l' [URL de base](#) du [document conteneur](#) du document .
2. Si l' [URL](#) du *document* est , et que l' URL de base du [créateur](#) du [contexte de navigation](#) du *document* n'est pas nulle, renvoyez cette [URL de base du créateur](#) [.about:blank](#)
3. Renvoie l'[URL](#) du *document* .

L' **URL de base du document** d'un [Document](#) objet est l' [enregistrement d'URL](#) obtenu en exécutant ces étapes :

1. Si aucun [base](#) élément n'a d' [href](#) attribut dans le [Document](#), renvoyez l' [URL de base de secours](#) [Document](#) du .
2. Sinon, renvoyez l' [URL de base gelée](#) du premier [base](#) élément du [Document](#) qui a un [href](#) attribut, dans l'[ordre de l'arborescence](#) .

Une **URL** **correspond** `about:blank` si son **schéma** est " `about`", son **chemin** contient une seule chaîne " `blank`", son **nom d'utilisateur** et **son mot de passe** sont la chaîne vide et son **hôte** est nul.

*La requête et le fragment d'une telle URL peuvent être non nuls. Par exemple, l'enregistrement d'URL créé en analysant " `about:blank?foo#bar` " correspond à `about:blank` .*

## 2.4.2 Analyser les URL

L'analyse d'une URL consiste à prendre une chaîne et à obtenir l'enregistrement d'URL qu'elle représente. Alors que ce processus est défini dans *URL* , la norme HTML définit un wrapper pour plus de commodité. [\[URL\]](#)

*Ce wrapper n'est utile que lorsque le codage de caractères pour l'analyseur d'URL doit correspondre à celui de l'objet de paramètres de document ou d'environnement pour des raisons d'héritage. Lorsque ce n'est pas le cas, l'analyseur d'URL peut être utilisé directement.*

Pour **analyser une URL** *url* , relative à un *document* ou à un *objet de paramètres d'environnement* , l'agent utilisateur doit suivre les étapes suivantes. L'analyse d'une URL entraîne soit un échec, soit une **chaîne d'URL résultante** et un **enregistrement d'URL résultant** .

1. Soit *encoding* le **codage de caractères** du *document* , si le *document* a été donné, et **le codage de caractères de l'URL** de l'API de l'objet des paramètres d'environnement dans le cas contraire.
2. Laissez *baseURL* être l' **URL de base** du *document* , si le *document* a été donné, et l' **URL de base de l' API** de l'objet des paramètres d'environnement sinon.
3. Soit *urlRecord* le résultat de l'application de **l'analyseur d'URL** à *url* , avec *baseURL* et *encoding* .
4. Si *urlRecord* est un échec, renvoie un échec.
5. Soit *urlString* le résultat de l'application du **sérialiseur d'URL** à *urlRecord* .
6. Renvoie *urlString* comme **chaîne d'URL résultante** et *urlRecord* comme **enregistrement d'URL résultant** .

## 2.4.3 Modifications dynamiques des URL de base



Lorsque l'[URL de base](#) d'un document change, tous les éléments de ce document sont [affectés par une modification de l'URL de base](#) .

Voici les [étapes de changement d'URL de base](#) , qui s'exécutent lorsqu'un élément est [affecté par un changement d'URL de base](#) (tel que défini par *DOM*) :

#### Si l'élément crée un [lien hypertexte](#)

Si l' [URL](#) identifiée par le lien hypertexte est présentée à l'utilisateur, ou si des données dérivées de cette [URL](#) affectent l'affichage, l' [href](#)attribut doit être [réanalysé](#) par rapport au [document de nœud](#) de l'élément et l'interface utilisateur mise à jour de manière appropriée.

Par exemple, les CSS [:link/](#) [:visited](#) [pseudo-classes](#) pourraient avoir été affectées.

Si le lien hypertexte a un [ping](#)attribut et que son ou ses [URL](#) sont affichés à l'utilisateur, les [ping](#)jetons de l'attribut doivent être [analysés](#) par rapport au [document de nœud](#) de l'élément et l'interface utilisateur mise à jour de manière appropriée.

#### Si l'élément est un élément [q](#), [blockquote](#), [ins](#) ou [del](#) avec un [cite](#)attribut

Si l' [URL](#) identifiée par l' [cite](#)attribut est présentée à l'utilisateur, ou si des données dérivées de cette [URL](#) affectent l'affichage, l' [URL](#) doit être [réanalysée](#) par rapport au [document de nœud](#) de l'élément et l'interface utilisateur mise à jour de manière appropriée.

#### Sinon

L'élément n'est pas directement affecté.

Par exemple, la modification de l'URL de base n'affecte pas l'image affichée par [img](#)les éléments, bien que les accès ultérieurs de l' [src](#)attribut IDL à partir du script renverront une nouvelle [URL absolue](#) qui pourrait ne plus correspondre à l'image affichée.

## 2.5 Récupérer des ressources

### 2.5.1 Terminologie

Une [réponse](#) dont le [type](#) est " `basic`", " `cors`" ou " `default`" est **CORS-same-origin** . [\[ALLER CHERCHER\]](#)

Une [réponse](#) dont le [type](#) est " `opaque`" ou " `opaqueredirect`" est **CORS-cross-origin** .

La réponse **non sécurisée** d' [une réponse](#) est sa [réponse interne](#) si elle en a une, et la [réponse](#) elle-même sinon.



Pour **créer une requête CORS potentielle** , en fonction d'une *url* , d'une *destination* , d'un *corsAttributeState* et d'un *indicateur de secours facultatif de même origine* , exécutez ces étapes :

1. Soit *mode* "no-cors" si *corsAttributeState* est [No CORS](#) , et " cors" sinon.
2. Si l'*indicateur de repli de même origine* est défini et que le *mode* est " no-cors", définissez le *mode* sur " same-origin".
3. Soit " *credentialsMode*include " ;
4. Si *corsAttributeState* est [Anonyme](#) , définissez le *mode d'accréditation* sur " same-origin".
5. Soit *request* une nouvelle [requête](#) dont l'[URL](#) est *url* , le [la destination](#) est *destination* , le [le mode](#) est *mode* , le [le mode d'informations d'identification](#) est *requirementsMode* et dont l'[l'indicateur use-URL-credentials](#) est défini.

## 2.5.2 Déterminer le type d'une ressource

Les **métadonnées Content-Type** d'une ressource doivent être obtenues et interprétées conformément aux exigences de *MIME Sniffing* . [\[MIMESNIFF\]](#)

Le [type MIME calculé](#) d'une ressource doit être trouvé d'une manière cohérente avec les exigences données dans *MIME Sniffing* . [\[MIMESNIFF\]](#)

Les [règles pour renifler spécifiquement les images](#) , les [règles pour distinguer si une ressource est textuelle ou binaire](#) , et les [règles pour renifler spécifiquement l'audio et la vidéo](#) sont également définies dans *MIME Sniffing* . Ces règles renvoient un [type MIME](#) comme résultat. [\[MIMESNIFF\]](#)

***Il est impératif que les règles de MIME Sniffing soient suivies à la lettre. Lorsqu'un agent utilisateur utilise des heuristiques différentes pour la détection du type de contenu par rapport à ce que le serveur attend, des problèmes de sécurité peuvent survenir. Pour plus de détails, voir MIME Sniffing . [\[MIMESNIFF\]](#)***

## 2.5.3 Extraction des encodages de caractères des metaéléments

L' **algorithme d'extraction d'un codage de caractères d'un metaélément** , étant donné une chaîne *s* , est le suivant. Il renvoie soit un encodage de caractères, soit rien.

1. Soit *position* un pointeur dans *s* , pointant initialement au début de la chaîne.
2. *Boucle* : recherche les sept premiers caractères dans *s* après *la position* qui correspondent à une correspondance [ASCII insensible à la casse](#) pour le mot " *charset*". Si aucune correspondance n'est trouvée, ne renvoie rien.
3. Ignorez tout [espace blanc ASCII](#) qui suit immédiatement le mot " *charset*" (il se peut qu'il n'y en ait pas).
4. Si le caractère suivant n'est pas un SIGNE ÉGAL U+003D (=), alors déplacez-vous *pour* pointer juste avant ce caractère suivant, et revenez à l'étape étiquetée *loop* .
5. Ignorez tout [espace blanc ASCII](#) qui suit immédiatement le signe égal (il se peut qu'il n'y en ait pas).
6. Traiter le caractère suivant comme suit :

**S'il s'agit d'un guillemet U+0022 (") et qu'il y a un guillemet U+0022 ultérieur (") dans *s***

**S'il s'agit d'un caractère U+0027 APOSTROPHE (') et qu'il y a un caractère U+0027 APOSTROPHE ultérieur (') dans *s***

Revoie le résultat de [l'obtention d'un encodage](#) à partir de la sous-chaîne située entre ce caractère et la prochaine occurrence la plus ancienne de ce caractère.

**S'il s'agit d'un guillemet U+0022 sans correspondance (")**

**S'il s'agit d'un caractère APOSTROPHE U+0027 sans correspondance (')**

**S'il n'y a pas de caractère suivant**

Ne rien retourner.

**Sinon**

Revoie le résultat de [l'obtention d'un codage](#) à partir de la sous-chaîne qui se compose de ce caractère jusqu'au premier [espace ASCII](#) ou caractère U+003B SEMICOLON (;), ou la fin de *s* , selon la première éventualité.

*Cet algorithme est distinct de ceux des spécifications HTTP (par exemple, HTTP n'autorise pas l'utilisation de guillemets simples et nécessite la prise en charge d'un mécanisme d'échappement antislash qui n'est pas pris en charge par cet algorithme). Alors que l'algorithme est utilisé dans des contextes qui, historiquement, étaient liés à HTTP, la syntaxe prise en charge par les implémentations a divergé il y a quelque temps. [\[HTTP\]](#)*

## 2.5.4 Attributs des paramètres CORS



Un **attribut de paramètres CORS** est un [attribut énuméré](#) . Le tableau suivant répertorie les mots-clés et les états de l'attribut — les états donnés dans la première cellule des lignes avec des mots-clés donnent les états auxquels ces mots-clés correspondent.

État	Mots clés	Brève description
Anonyme	<b>anonymous</b>	Les <a href="#">requêtes</a> pour l'élément auront leur <a href="#">mode</a> défini sur " cors " et leur <a href="#">mode d'identification</a> défini sur " same-origin ".
	(la chaîne vide)	
Utiliser les informations d'identification	<b>use-credentials</b>	Les <a href="#">requêtes</a> pour l'élément auront leur <a href="#">mode</a> défini sur " cors " et leur <a href="#">mode d'identification</a> défini sur " include ".

La [valeur par défaut non valide](#) de l'attribut est l' état [Anonyme](#) et sa [valeur par défaut manquante](#) est l' état **Pas de CORS** . Pour les besoins de [la réflexion](#) , le [mot-clé canonique](#) de l' état [Anonyme](#) est le [anonymous](#) mot-clé.

La majorité des récupérations régies par [les attributs des paramètres CORS](#) seront effectuées via l' algorithme [de création d'une requête CORS potentielle](#) .

Pour les fonctionnalités plus modernes, où le [mode](#) de la demande est toujours " cors ", certains [attributs de paramètres CORS](#) ont été réorientés pour avoir une signification légèrement différente, dans laquelle ils n'affectent que le [mode d'identification](#) de la [demande](#) . Pour effectuer cette traduction, nous définissons le **mode d'informations d'identification de l'attribut de paramètres CORS** pour un [attribut de paramètres CORS](#) donné à déterminer en activant l'état de l'attribut :

### Pas de CORS

#### Anonyme

" same-origin "

#### Utiliser les informations d'identification

" include "

## 2.5.5 Attributs de la politique de référence

Un **attribut de stratégie de référence** est un [attribut énuméré](#) . Chaque [stratégie de référence](#) , y compris la chaîne vide, est un mot-clé pour cet attribut, mappé à un état du même nom.

La [valeur par défaut non valide](#) et [la valeur par défaut manquante](#) de l'attribut sont toutes deux à l'état de chaîne vide.

L'impact de ces états sur le modèle de traitement des différentes [extractions](#) est défini plus en détail tout au long de cette spécification, dans *Fetch* et dans *Referrer Policy* . [\[RÉCUPÉRER\]](#) [\[POLITIQUE DE RÉFÉRENCE\]](#)

*Plusieurs signaux peuvent contribuer au modèle de traitement utilisé pour un [fetch](#) donné ; un [attribut de stratégie de référence](#) n'est que l'un d'entre eux. En général, l'ordre dans lequel ces signaux sont traités est :*

1. *Tout d'abord, la présence d'un [noreferrertype](#) de lien ;*
2. *Ensuite, la valeur d'un [attribut de politique de référence](#) ;*
3. *Ensuite, la présence de tout [meta](#)élément dont [name](#) l'attribut est défini sur [referrer](#).*
4. *Enfin, l'[Referrer-Policy](#) en-tête HTTP ``.*

## 2.5.6 Attributs nonce

✓ MDN

Un **nonce** attribut de contenu représente un nonce cryptographique ("nombre utilisé une fois") qui peut être utilisé par la *politique de sécurité du contenu* pour déterminer si une récupération donnée sera autorisée ou non à se poursuivre. La valeur est du texte. [\[CSP\]](#)

Les éléments qui ont un **nonce** attribut de contenu garantissent que le nonce cryptographique n'est exposé qu'au script (et non aux canaux secondaires comme les sélecteurs d'attributs CSS) en prenant la valeur de l'attribut de contenu, en le déplaçant dans un emplacement interne nommé `[[CryptographicNonce]]` , en l'exposant au script via l' [HTMLOrSVGElement](#) interface mixin et en définissant l'attribut content sur la chaîne vide. Sauf indication contraire, la valeur de l'emplacement est la chaîne vide.

**`element.nonce`**

Renvoie la valeur définie pour le nonce cryptographique de l' *élément* . Si le setter n'a pas été utilisé, ce sera la valeur trouvée à l'origine dans l' **nonce** attribut content.

**`element.nonce`** = *value*

Met à jour la valeur nonce cryptographique de l'*élément* .

MDN

L'attribut IDL doit, lors de l'obtention, renvoyer la valeur du `[[CryptographicNonce]]` **nonce** de cet élément ; [et lors du réglage, définissez le `\[\[CryptographicNonce\]\]`](#) de cet élément sur la valeur donnée.

Notez que le setter de l' nonce attribut IDL ne met pas à jour l'attribut de contenu correspondant. Ceci, ainsi que le réglage ci-dessous de l' nonce attribut de contenu sur la chaîne vide lorsqu'un élément devient un contexte de navigation connecté , vise à empêcher l'exfiltration de la valeur nonce via des mécanismes qui peuvent facilement lire les attributs de contenu, tels que les sélecteurs. Apprenez-en plus dans le numéro 2369 , où ce comportement a été introduit.

Les étapes de modification d'attribut suivantes sont utilisées pour l' nonce attribut de contenu :

1. Si l'élément n'inclut pas HTMLOrSVGELEMENT, alors retournez.
2. Si *localName* n'est pas nonce ou si l'espace de noms n'est pas nul, alors retournez.
3. Si la valeur est nulle, définissez [[CryptographicNonce]] de l' élément sur la chaîne vide.
4. Sinon, définissez [[CryptographicNonce]] de l' élément sur la valeur .

Chaque fois qu'un élément incluant HTMLOrSVGELEMENT devient connecté au contexte de navigation , l'agent utilisateur doit exécuter les étapes suivantes sur l' élément :

1. Soit CSP list la liste CSP shadow de l'élément , y compris la politique du conteneur racine .
2. Si la liste CSP contient une politique de sécurité du contenu fournie par l'en-tête et que l'élément a un nonce attribut de contenu *attr* dont la valeur n'est pas la chaîne vide, alors :
  1. Soit *nonce* le [[CryptographicNonce]] de l' élément .
  2. Définissez une valeur d'attribut pour l'élément en utilisant " nonce " et la chaîne vide.
  3. Définissez [[CryptographicNonce]] de l' élément sur *nonce* .

*Si [[CryptographicNonce]] de l' élément n'était pas restauré, ce serait la chaîne vide à ce stade.*

Les étapes de clonage pour les éléments qui incluent HTMLOrSVGELEMENT doivent définir l' emplacement [[CryptographicNonce]] sur la copie à la valeur de l'emplacement sur l'élément en cours de clonage.

## 2.5.7 Attributs de chargement différé

Un **attribut de chargement différé** est un [attribut énuméré](#) . Le tableau suivant répertorie les mots-clés et les états de l'attribut — les mots-clés de la colonne de gauche correspondent aux états de la cellule de la deuxième colonne sur la même ligne que le mot-clé.

L'attribut ordonne à l'agent utilisateur de récupérer une ressource immédiatement ou de différer la récupération jusqu'à ce que certaines conditions associées à l'élément soient remplies, selon l'état actuel de l'attribut.

Mot-clé	État	Description
<b>lazy</b>	<b>Paresseux</b>	Utilisé pour différer la récupération d'une ressource jusqu'à ce que certaines conditions soient remplies.
<b>eager</b>	<b>Désireux</b>	Utilisé pour récupérer une ressource immédiatement ; l'état par défaut.

La [valeur par défaut manquante](#) et [la valeur par défaut non valide](#) de l'attribut sont toutes deux à l'état [Impatient](#) .

Les **étapes de chargement paresseux de l'élément** , étant donné un élément element , sont les suivantes :

1. Si [le script est désactivé](#) pour l'élément , renvoie false.

*Il s'agit d'une mesure anti-pistage, car si un agent utilisateur prenait en charge le chargement différé lorsque le script est désactivé, il serait toujours possible pour un site de suivre la position de défilement approximative d'un utilisateur tout au long d'une session, en plaçant stratégiquement des images dans le balisage d'une page de sorte que un serveur peut suivre le nombre d'images demandées et à quel moment.*

2. Si [l'attribut de chargement paresseux](#) de l'élément est dans l'état [Lazy](#) , alors retourne true.
3. Renvoie faux.

Chaque élément [img](#) et [iframe](#) est associé à **des étapes de reprise de chargement paresseux** , initialement nulles.

*Pour les éléments [img](#) qui [effectueront un chargement différé](#) , ces étapes sont exécutées à partir du rappel de [l'observateur d'intersection de chargement différé](#) ou*

lorsque leur attribut de chargement différé est défini sur l'état Eager. Cela entraîne la poursuite du chargement de l'élément iframe

Chacun Document a un **observateur d'intersection de chargement différé**, initialement défini sur null mais peut être défini sur une IntersectionObserver instance.

Pour **commencer à observer l'intersection d'un élément** à chargement différé element, exécutez ces étapes :

1. Soit *doc* le nœud document de l'élément.
2. Si l'observateur d'intersection de chargement paresseux de *doc* est nul, définissez-le sur une nouvelle instance, initialisée comme suit : IntersectionObserver

L'intention est d'utiliser la valeur d'origine du IntersectionObserver constructeur. Cependant, nous sommes obligés d'utiliser le constructeur exposé à JavaScript dans cette spécification, jusqu'à ce qu'*Intersection Observer* expose des crochets de bas niveau à utiliser dans les spécifications. Voir le bogue [w3c/IntersectionObserver#464](https://bugzilla.mozilla.org/show_bug.cgi?id=1429282) qui suit cela. [\[OBSERVATEUR D'INTERSECTION\]](#)

- Le *rappel* est ces étapes, avec des entrées d'arguments et un observateur :

1. Pour chaque *entrée* dans les entrées **utilisant une méthode d'itération qui ne déclenche pas d'accesseurs de tableau modifiables par le développeur ou de crochets d'itération** :

1. Laissez *ResumptionSteps* être nul.
2. Si *entrée* . isIntersecting est vrai, puis définissez *resumptionSteps* sur *entry* . étapes de reprise de chargement paresseux target de .
3. Si *ResumptionSteps* est null, alors retournez.
4. Arrêtez d'observer l'intersection d'un élément de chargement paresseux pour l'entrée . target.
5. Définir l'entrée . étapes de reprise de chargement paresseux target à null.
6. Appelez *resumptionSteps* .



L'intention est d'utiliser la valeur d'origine des getters `isIntersecting` et `target`. Voir [w3c/IntersectionObserver#464](#) . [OBSERVATEUR D'INTERSECTION]

- Les *options* sont un `IntersectionObserverInit` dictionnaire avec les membres de dictionnaire suivants : « [ " `rootMargin`" → [lazy load root margin](#) ] »

*Cela permet de récupérer l'image pendant le défilement, lorsqu'elle ne croise pas encore - mais est sur le point de - croiser la fenêtre d'affichage.*

Les suggestions [de marge racine de chargement différé](#) impliquent des modifications dynamiques de la valeur, mais l' `IntersectionObserver` API ne prend pas en charge la modification de la marge racine. Voir le problème [w3c/IntersectionObserver#428](#) .

3. Appelez la méthode [de lazy load intersection observer](#) de *doc* avec *element* comme argument. `observe`

L'intention est d'utiliser la valeur d'origine de la `observe` méthode. Voir [w3c/IntersectionObserver#464](#) . [OBSERVATEUR D'INTERSECTION]

Pour **arrêter l'observation d'intersection d'un élément de chargement différé** *element* , exécutez ces étapes :

1. Soit *doc* le [nœud document](#) de l' *élément* .
2. [Assert](#) : l'[observateur d'intersection de chargement paresseux](#) de *doc* n'est pas nul.
3. Appelez la méthode [d'observateur d'intersection de chargement paresseux](#) de *doc* avec *element* comme argument. `unobserve`

L'intention est d'utiliser la valeur d'origine de la `unobserve` méthode. Voir [w3c/IntersectionObserver#464](#) . [OBSERVATEUR D'INTERSECTION]

La **marge racine de chargement différé** est une valeur [définie par l'implémentation](#) , mais avec les suggestions suivantes à prendre en compte :



- Définissez une valeur minimale qui entraîne le plus souvent le chargement des ressources avant qu'elles ne croisent la fenêtre d'affichage dans des modèles d'utilisation normaux pour le périphérique donné.
- La vitesse de défilement typique : augmentez la valeur pour les appareils avec des vitesses de défilement typiques plus rapides.
- La vitesse ou l'élan de défilement actuel : l'UA peut tenter de prédire où le défilement s'arrêtera probablement et ajuster la valeur en conséquence.
- La qualité du réseau : augmentez la valeur pour les connexions lentes ou à latence élevée.
- Les préférences de l'utilisateur peuvent influencer la valeur.

*Il est important pour la confidentialité que la marge racine de chargement différé ne divulgue pas d'informations supplémentaires. Par exemple, la vitesse de défilement typique sur l'appareil actuel pourrait être imprécise afin de ne pas introduire de nouveau vecteur d'empreinte.*

## 2.5.8 Attributs de blocage

Un **attribut de blocage** indique explicitement que certaines opérations doivent être bloquées lors de la récupération d'une ressource externe. Les opérations pouvant être bloquées sont représentées par **des jetons de blocage possibles**, qui sont des chaînes répertoriées dans le tableau suivant :

Jeton de blocage possible	Description
"render"	L'élément est <u>potentiellement bloquant le rendu</u> .

*À l'avenir, il pourrait y avoir plus de jetons de blocage possibles.*

Un attribut de blocage doit avoir une valeur qui est un ensemble non ordonné de jetons uniques séparés par des espaces, dont chacun est un jeton de blocage possible. Les jetons pris en charge d'un attribut de blocage sont les jetons de blocage possibles. Tout élément peut avoir au plus un attribut bloquant.

Les **jetons de blocage définis** pour un élément *eI* sont le résultat des étapes suivantes :

1. Soit *value* la valeur de l'attribut bloquant de *eI*, ou la chaîne vide si aucun attribut de ce type n'existe.
2. Définissez *la valeur* sur *valeur*, convertie en minuscules ASCII.
3. Soit *rawTokens* le résultat de la division de la valeur sur l'espace blanc ASCII.

4. Renvoie un ensemble contenant les éléments de *rawTokens* qui sont [des jetons de blocage possibles](#) .

Un élément est **potentiellement bloquant le rendu** si son [jeu de jetons de blocage](#) contient "[render](#)", ou s'il est **implicitement potentiellement bloquant le rendu** , ce qui sera défini au niveau des éléments individuels. Par défaut, un élément n'est pas [implicitement potentiellement bloquant le rendu](#) .

## 2.6 Interfaces DOM communes

### 2.6.1 Refléter les attributs de contenu dans les attributs IDL

Certains attributs IDL sont définis pour **réfléter** un attribut de contenu particulier. Cela signifie qu'à l'obtention, l'attribut IDL renvoie la valeur actuelle de l'attribut de contenu, et lors de la définition, l'attribut IDL change la valeur de l'attribut de contenu à la valeur donnée.

En général, lors de l'obtention, si l'attribut content n'est pas présent, l'attribut IDL doit agir comme si la valeur de l'attribut content était la chaîne vide ; et lors de la définition, si l'attribut de contenu n'est pas présent, il doit d'abord être ajouté.

Les attributs IDL de type [DOMString](#) ou [DOMString?](#) qui reflètent des attributs de contenu [énumérés](#) peuvent être **limités aux seules valeurs connues** . Selon les modèles de traitement ci-dessous, ceux-ci feront que les getters pour ces attributs IDL ne renverront que des mots-clés pour ces attributs énumérés, ou la chaîne vide ou null.

Si un attribut IDL réfléchissant a le type [DOMString](#):

- Les étapes du getter sont :
  1. Si l'attribut content est un [attribut énuméré](#) et que l'attribut IDL est [limité aux seules valeurs connues](#) :
    1. Si l'attribut de contenu n'est dans aucun état (par exemple, l'attribut est manquant et qu'il n'y a pas [de valeur manquante default](#) ), ou si l'attribut de contenu est dans un état sans valeur de mot-clé associée, renvoyez la chaîne vide.
    2. Renvoie le [mot-clé canonique](#) pour l'état de l'attribut de contenu.
  2. Sinon:
    1. Renvoie la valeur de l'attribut de contenu.
- Les étapes du setter consistent à définir la valeur de l'attribut de contenu sur la valeur donnée.

Si un attribut IDL réfléchissant a le type DOMString?:

- Les étapes du getter sont :
  1. Assert : l'attribut content est un attribut énuméré .
  2. Assert : l'attribut IDL est limité aux seules valeurs connues .
  3. Assert : l'attribut de contenu est dans un certain état.
  4. Si l'attribut de contenu est dans un état sans valeur de mot-clé associée, alors renvoie null.
  5. Renvoie le mot-clé canonique pour l'état de l'attribut de contenu.
- Les étapes du setter sont :
  1. Si la valeur donnée est nulle, supprimez l'attribut de contenu.
  2. Sinon, définissez la valeur de l'attribut de contenu sur la valeur donnée.

Si un attribut IDL réfléchissant a le type USVString:

- Les étapes du getter sont :
  1. Si l'attribut content est défini pour contenir une URL :
    1. Si l'attribut de contenu est absent, renvoyez la chaîne vide.
    2. Analysez la valeur de l'attribut content par rapport au nœud document de l'élément .
    3. Si cela réussit, renvoyez la chaîne d'URL résultante .
    4. Sinon, retournez la valeur de l'attribut content, convertie en USVString.
  2. Sinon:
    1. Renvoie la valeur de l'attribut content, convertie en USVString.
- Les étapes du setter consistent à définir la valeur de l'attribut de contenu sur la valeur donnée.

Si un attribut IDL réfléchissant est un boolean attribut, alors lors de l'obtention de l'attribut IDL, il doit renvoyer vrai si l'attribut de contenu est défini, et faux s'il est absent. Lors de la définition, l'attribut content doit être supprimé si l'attribut IDL est défini sur false, et doit être défini sur la chaîne vide si l'attribut IDL est défini sur true. (Cela correspond aux règles pour les attributs de contenu booléen .)

Si un attribut IDL réfléchissant a un type entier signé ( [long](#) ) alors, lors de l'obtention, l'attribut content doit être analysé selon les [règles d'analyse des entiers signés](#) , et si cela réussit, et que la valeur est dans la plage du type de l'attribut IDL , la valeur résultante doit être renvoyée. Si, au contraire, il échoue ou renvoie une valeur hors plage, ou si l'attribut est absent, alors la valeur par défaut doit être renvoyée à la place, ou 0 s'il n'y a pas de valeur par défaut. Lors de la définition, la valeur donnée doit être convertie en la chaîne la plus courte possible représentant le nombre sous la forme d'un [entier valide](#) , puis cette chaîne doit être utilisée comme nouvelle valeur d'attribut de contenu.

Si un attribut IDL réfléchissant a un type entier signé ( [long](#) ) qui est **limité aux seuls nombres non négatifs** alors, lors de l'obtention, l'attribut de contenu doit être analysé conformément aux [règles d'analyse des entiers non négatifs](#) , et si cela réussit, et la valeur est dans la plage du type de l'attribut IDL, la valeur résultante doit être renvoyée. Si, au contraire, il échoue ou renvoie une valeur hors plage, ou si l'attribut est absent, la valeur par défaut doit être renvoyée à la place, ou -1 s'il n'y a pas de valeur par défaut. Lors du paramétrage, si la valeur est négative, l'agent utilisateur doit lancer un ["IndexSizeError" DOMException](#) . Sinon, la valeur donnée doit être convertie en la chaîne la plus courte possible représentant le nombre sous forme de [entier non négatif valide](#) , puis cette chaîne doit être utilisée comme nouvelle valeur d'attribut de contenu.

Si un attribut IDL réfléchissant a un type entier *non signé* [unsigned long](#) ( ) alors, lors de l'obtention, l'attribut de contenu doit être analysé conformément aux [règles d'analyse des entiers non négatifs](#) , et si cela réussit, et que la valeur est comprise entre 0 et 2147483647 inclus, la valeur résultante doit être renvoyée. Si, au contraire, il échoue ou renvoie une valeur hors plage, ou si l'attribut est absent, la valeur par défaut doit être renvoyée à la place, ou 0 s'il n'y a pas de valeur par défaut. Au réglage, d'abord, si la nouvelle valeur est comprise entre 0 et 2147483647, alors soit *n* la nouvelle valeur, sinon soit *n* la valeur par défaut, ou 0 s'il n'y a pas de valeur par défaut ; alors, *n* doit être convertie en la chaîne la plus courte possible représentant le nombre sous la forme d'un [entier non négatif valide](#) et cette chaîne doit être utilisée comme nouvelle valeur d'attribut de contenu.

Si un attribut IDL réfléchissant a un type entier non signé ( [unsigned long](#) ) qui est **limité aux seuls nombres non négatifs supérieurs à zéro** , alors le comportement est similaire au cas précédent, mais zéro n'est pas autorisé. Lors de l'obtention, l'attribut content doit d'abord être analysé conformément aux [règles d'analyse des entiers non négatifs](#) , et si cela réussit et que la valeur est comprise entre 1 et 2147483647 inclus, la valeur résultante doit être renvoyée. Si, au contraire, il échoue ou renvoie une valeur hors plage, ou si l'attribut est absent, la valeur par défaut doit être renvoyée à la place, ou 1 s'il n'y a pas de valeur par défaut. Au réglage, si la valeur est zéro, l'agent utilisateur doit lancer un ["IndexSizeError" DOMException](#) . Sinon, d'abord, si la nouvelle valeur est comprise entre 1 et 2147483647, alors soit *n* la nouvelle valeur, sinon soit *n* la valeur par défaut, ou 1 s'il n'y a pas de valeur par défaut ; ensuite, *n* doit être converti en la chaîne la plus courte possible représentant le nombre sous la forme d'un [entier non négatif valide](#) et cette chaîne doit être utilisée comme nouvelle valeur d'attribut de contenu.

Si un attribut IDL réfléchissant a un type entier non signé ( [unsigned long](#)) qui est **limité aux seuls nombres non négatifs supérieurs à zéro avec fallback** , alors le comportement est similaire au cas précédent, mais les valeurs non autorisées sont converties à la valeur par défaut. Lors de l'obtention, l'attribut content doit d'abord être analysé conformément aux [règles d'analyse des entiers non négatifs](#) , et si cela réussit et que la valeur est comprise entre 1 et 2147483647 inclus, la valeur résultante doit être renvoyée. Si, en revanche, il échoue ou renvoie une valeur hors plage, ou si l'attribut est absent, la valeur par défaut doit être renvoyée à la place. Au réglage, d'abord, si la nouvelle valeur est comprise entre 1 et 2147483647, alors soit  $n$  la nouvelle valeur, sinon soit  $n$  soit la valeur par défaut ; ensuite,  $n$  doit être converti en la chaîne la plus courte possible représentant le nombre sous la forme d'un [entier non négatif valide](#) et cette chaîne doit être utilisée comme nouvelle valeur d'attribut de contenu.

Si un attribut IDL réfléchissant a un type entier non signé ( [unsigned long](#)) qui est **limité à la plage** [  $min$  ,  $max$  ], alors lors de l'obtention, l'attribut content doit d'abord être analysé selon les [règles d'analyse des entiers non négatifs](#) , et si c'est le cas réussie et que la valeur est comprise entre  $min$  et  $max$  inclus, la valeur résultante doit être renvoyée. En cas d'échec, la valeur par défaut doit être renvoyée. S'il réussit mais que la valeur est inférieure à  $min$  ,  $min$  doit être renvoyé. S'il réussit mais que la valeur est supérieure à  $max$  ,  $max$  doit être retourné. Lors de la définition, il se comporte de la même manière que la définition d'un entier non signé reflété régulier.

Si un attribut IDL réfléchissant a un type de nombre à virgule flottante ( [double](#) ou [unrestricted double](#)), alors, lors de l'obtention, l'attribut de contenu doit être analysé conformément aux [règles d'analyse des valeurs de nombre à virgule flottante](#) , et si cela réussit, la valeur résultante doit être revenu. Si au contraire il échoue, ou si l'attribut est absent, il faut retourner la valeur par défaut à la place, ou 0.0 s'il n'y a pas de valeur par défaut. Lors de la définition, la valeur donnée doit être convertie en la [meilleure représentation du nombre sous forme de nombre à virgule flottante](#) , puis cette chaîne doit être utilisée comme nouvelle valeur d'attribut de contenu.

Si un attribut IDL réfléchissant a un type de nombre à virgule flottante ( [double](#) ou [unrestricted double](#)) qui est **limité aux nombres supérieurs à zéro** , alors le comportement est similaire au cas précédent, mais les valeurs nulles et négatives ne sont pas autorisées. Lors de l'obtention, l'attribut content doit être analysé conformément aux [règles d'analyse des valeurs de nombre à virgule flottante](#) , et si cela réussit et que la valeur est supérieure à 0,0, la valeur résultante doit être renvoyée. Si, en revanche, il échoue ou renvoie une valeur hors plage, ou si l'attribut est absent, la valeur par défaut doit être renvoyée à la place, ou 0,0 s'il n'y a pas de valeur par défaut. Au réglage, si la valeur est inférieure ou égale à zéro, alors la valeur doit être ignorée. Sinon, la valeur donnée doit être convertie en [meilleure représentation du nombre sous forme de nombre à virgule flottante](#) , puis cette chaîne doit être utilisée comme nouvelle valeur d'attribut de contenu.

*Les valeurs Infinity et Not-a-Number (NaN) lèvent une exception lors du paramétrage, comme défini dans Web IDL . [\[WEBIDL\]](#)*

Si un attribut IDL réfléchissant a le type [DOMTokenList](#), alors lors de son obtention, il doit renvoyer un [DOMTokenList](#) objet dont l'élément associé est l'élément en question et dont le nom local de l'attribut associé est le nom de l'attribut en question.

Si un attribut IDL réfléchissant *attr* a le type *T?*, où *T* est soit [Element](#) soit une interface qui hérite de [Element](#), alors :

- Les éléments du type sur lequel cet attribut IDL apparaît ont un ***attr - élément explicitement défini***, qui est une référence faible à un élément ou nul. Il est initialement nul.
- Les éléments du type sur lequel cet attribut IDL apparaît ont un ***élément associé à attr***. Pour calculer l' [élément attr](#) -associated pour un tel élément *element* :
  1. Si l' [élément explicitement défini attr -element](#) n'est pas nul :
    - Si l' [élément attr explicitement défini de l' élément](#) est un [descendant](#) de l'un des ancêtres de l' [élément](#) , y compris l' [ombre](#) , alors renvoie l' [élément attr explicitement défini de l' élément](#) .
    - Sinon, renvoie null.
  2. Sinon, si l'attribut content est présent sur *element*, alors renvoyez le premier élément *candidat*, dans [l'ordre de l'arborescence](#), qui répond aux critères suivants :
    - [la racine](#) du *candidat* est la même que [la racine](#) de l' [élément](#) ,
    - [l'ID](#) du *candidat* est la valeur de l'attribut de contenu, et
    - le *candidat* [implémente](#) *T* .

Si aucun élément de ce type n'existe, renvoie null.

3. Renvoie nul.

*D'autres parties de cette spécification, ou d'autres spécifications utilisant la réflexion d'attribut, sont supposées consulter l'élément [attr -associated](#) d'un élément . L'élément [attr explicitement défini d' un élément](#) est un détail d'implémentation interne de son [élément associé à attr](#) et ne doit pas être utilisé directement.*

- Les étapes du getter consistent à renvoyer [cet](#) élément [associé à attr](#) .
- Les étapes du setter sont :
  1. Si la valeur donnée est nulle, alors :
    - Définissez [explicitement cet](#) élément [attr sur](#) null.
    - Supprimez l'attribut content de [ce fichier](#) .

- Retour.
2. Définissez la valeur de l'attribut de contenu pour [this](#) sur la chaîne vide.
  3. Définissez [explicitement cet](#) élément [attr sur une référence faible à la valeur](#) donnée.
- [Les étapes de changement d'attribut](#) suivantes , étant donné *element* , *localName* , *oldValue* , *value* et *namespace* , sont utilisées pour synchroniser entre l'attribut content et l'attribut IDL :
    1. Si *localName* n'est pas le nom local de l'attribut de contenu, ou si *l'espace de noms* n'est pas nul, alors retournez.
    2. Définissez [explicitement attr -element](#) de *l'élément* sur null.

Si un attribut IDL réfléchissant *attr* a le type , où *T* est soit soit une interface qui hérite de , alors `:FrozenArray<T>?ElementElement`

- Les éléments du type sur lequel cet attribut IDL apparaît ont **explicitement défini attr -elements** , qui est soit une [liste](#) de références faibles aux éléments, soit null. Il est initialement nul.
- Les éléments du type sur lequel cet attribut IDL apparaît ont **des éléments associés à attr mis en cache** , qui est un `nul.FrozenArray<T>?` . Il est initialement nul.
- Les éléments du type sur lequel cet attribut IDL apparaît ont **des éléments associés à attr** . Pour calculer les [éléments associés à attr](#) pour un tel *élément element* :
  1. Soit *elements* une [liste](#) vide .
  2. Si l' *élément* [explicitement défini attr -elements](#) n'est pas nul, alors :
    1. [Pour chaque attrElement](#) dans les *attr -elements* [explicitement définis de l'élément](#) :
      1. Si *attrElement* n'est un [descendant](#) d'aucun des [ancêtres shadow-inclusives](#) de *element* , alors [continuez](#) .
      2. [Ajoutez attrElement](#) aux *éléments* .
  3. Sinon:
    1. Si l'attribut content n'est pas présent sur *element* , renvoyez null.
    2. Soit *jetons* la valeur de l'attribut de contenu, [divisée en espaces blancs ASCII](#) .
    3. [Pour chaque identifiant](#) dans *les jetons* :



1. Soit *candidat* le premier élément, dans l'ordre de l'arborescence , qui répond aux critères suivants :
  - la racine du *candidat* est la même que la racine de l'élément ,
  - l' ID du *candidat* est *id* , et
  - le *candidat* implémente *T* .

Si aucun élément de ce type n'existe, continuez .

2. Ajouter un *candidat* aux éléments .

4. Éléments de retour .

*D'autres parties de cette spécification, ou d'autres spécifications utilisant la réflexion d'attribut, sont censées consulter les éléments associés à attr d'un élément . Les éléments attr définis explicitement d'un élément sont un détail d'implémentation interne de ses éléments associés à attr et ne doivent pas être utilisés directement. De même, les éléments attr -associated en cache de l'élément sont un détail d'implémentation interne du getter de l'attribut IDL.*

- Les étapes du getter sont :

1. Soit *elements* les éléments associés à attr de this .
2. Si le contenu des *elements* est égal au contenu des éléments associés à l' attr en cache de this , alors renvoie les éléments associés à l' attr en cache de this .
3. Soit *elementsAsFrozenArray* être *elements* , convertis en `a.FrozenArray<T>?`
4. Définissez ces éléments associés à l' attr mis en cache sur *elementsAsFrozenArray* .
5. Retourne *elementsAsFrozenArray* .

*Cette couche de mise en cache supplémentaire est nécessaire pour préserver l'invariant que `element.reflectedElements === element.reflectedElements`.*

- Les étapes du setter sont :

1. Si la valeur donnée est nulle :
  1. Définissez explicitement ces éléments *attr* sur `null` .
  2. Supprimez l'attribut contenu de ce fichier .
  3. Retour.
2. Définissez la valeur de l'attribut de contenu pour this sur la chaîne vide.
3. Soit *elements* une liste vide .



4. [Pour chaque](#) *élément* de la valeur donnée :
  1. [Ajouter](#) une référence faible à *element* à *elements* .
5. Définissez [cet](#) ensemble [explicitement attr -elements](#) sur *elements* .
- [Les étapes de changement d'attribut](#) suivantes , étant donné *element* , *localName* , *oldValue* , *value* et *namespace* , sont utilisées pour synchroniser entre l'attribut contenu et l'attribut IDL :
  1. Si *localName* n'est pas le nom local de l'attribut de contenu, ou si *l'espace de noms* n'est pas nul, alors retournez.
  2. Définissez [explicitement attr -elements](#) de *l'élément* sur null.

## 2.6.2 Collectes

Les interfaces [HTMLFormControlsCollection](#) et sont [des collections](#) dérivées de l' interface. L' interface est une [collection](#) , mais n'est pas si dérivée. [HTMLOptionsCollectionHTMLCollectionHTMLAllCollection](#)

### 2.6.2.1 L' [HTMLAllCollection](#) interface

L' [HTMLAllCollection](#) interface est utilisée pour l' [document.all](#) attribut hérité. Il fonctionne de manière similaire à [HTMLCollection](#); les principales différences sont qu'il permet une variété stupéfiante d'utilisations (ab) différentes de ses méthodes pour finir par retourner quelque chose, et qu'il peut être appelé comme une fonction comme alternative à l'accès à la propriété.

*Tous [HTMLAllCollection](#) les objets sont enracinés à a [Document](#) et ont un filtre qui correspond à tous les éléments, de sorte que les éléments [représentés par la collection](#) d'un [HTMLAllCollection](#) objet se composent de tous les éléments descendants de la racine [Document](#).*

Les objets qui implémentent l' [HTMLAllCollection](#) interface sont [des objets de plate-forme hérités](#) avec une méthode interne `[[Call]]` supplémentaire décrite dans la [section ci-dessous](#) . Ils ont également un emplacement interne `[[IsHTMLDDA]]` .

*Les objets qui implémentent l' [HTMLAllCollection](#) interface ont plusieurs comportements inhabituels, dus au fait qu'ils ont un slot interne `[[IsHTMLDDA]]` :*

- *L' opération abstraite [ToBoolean](#) en JavaScript renvoie false lorsqu'on donne des objets implémentant l' [HTMLAllCollection](#) interface.*

- L'opération abstraite [IsLooselyEqual](#), lorsqu'elle donne des objets implémentant l' [HTMLAllCollection](#) interface, renvoie true lorsqu'elle est comparée aux valeurs `undefined` et `null` (Les comparaisons utilisant l'opération abstraite [IsStrictlyEqual](#) et les comparaisons [IsLooselyEqual](#) avec d'autres valeurs telles que des chaînes ou des objets ne sont pas affectées.)
- L' `typeof` opérateur en JavaScript renvoie la chaîne "undefined" lorsqu'il est appliqué aux objets implémentant l' [HTMLAllCollection](#) interface.

Ces comportements spéciaux sont motivés par un désir de compatibilité avec deux classes de contenu hérité : une qui utilise la présence de [document.all](#) comme moyen de détecter les agents utilisateurs hérités, et une qui ne prend en charge que ces agents utilisateurs hérités et utilise l' [document.all](#) objet sans tester sa présence. d'abord. [\[JAVASCRIPT\]](#)

```
[Exposed=Window,

LegacyUnenumerableNamedProperties]

interface HTMLAllCollection {

    readonly attribute unsigned long length;

    getter Element (unsigned long index);

    getter (HTMLCollection or Element)? namedItem(DOMString name);

    (HTMLCollection or Element)? item(optional DOMString
nameOrIndex);

    // Note: HTMLAllCollection objects have a custom \[\[Call\]\]
internal method and an \[\[IsHTMLDDA\]\] internal slot.

};
```

Les [index de propriété pris en charge](#) par l'objet sont tels que définis pour [HTMLCollection](#) les objets.

Les [noms de propriété pris en charge](#) sont constitués des valeurs non vides de tous les [id](#) attributs de tous les éléments [représentés par la collection](#), et des valeurs non vides de tous les [name](#) attributs de tous les [éléments nommés "all" représentés par la collection](#), dans [tree order](#), en ignorant les doublons ultérieurs, avec le [id](#) d'un élément précédant son [name](#) s'il contribue à la fois, ils diffèrent l'un de l'autre, et aucun n'est le doublon d'une entrée antérieure.

Les **length** étapes du getter consistent à renvoyer le nombre de nœuds représentés par la collection .

Le getter de la propriété indexée doit renvoyer le résultat de l'obtention de l'élément indexé "all" à partir de this compte tenu de l'index passé.

Les étapes de la méthode consistent à renvoyer le résultat de l'obtention du ou des éléments nommés "tous" à partir de ce *nom* donné **.namedItem(name)**

Les étapes de la méthode sont : **item(nameOrIndex)**

1. Si *nameOrIndex* n'a pas été fourni, renvoie null.
  2. Renvoie le résultat de l'obtention du ou des éléments indexés ou nommés "tous" à partir de this , étant donné *nameOrIndex* .
- 

Les éléments suivants sont **des éléments nommés**

**"tous"** : a, button, embed, form, frame, frameset, iframe, img, input, map, meta, object, select, et textarea

Pour **obtenir l'élément indexé**

**"tous"** d'une HTMLAllCollection *collection* donnée un index *index* , renvoyez l' *index*<sup>th</sup> élément dans *collection* , ou null s'il n'y a pas un tel *index*<sup>th</sup> élément.

Pour **obtenir le ou les éléments nommés**

**"tous"** d'une HTMLAllCollection *collection* portant le nom *name* , procédez comme suit :

1. Si *name* est la chaîne vide, renvoie null.
2. Soit *subCollection* un HTMLCollection objet ayant la même racine Document que *collection* , dont le filtre correspond uniquement aux éléments qui sont soit :
  - "tous" les éléments nommés avec un *name* attribut égal à *name* , ou,
  - éléments avec un ID égal à *name* .
3. S'il y a exactement un élément dans *subCollection* , renvoyez cet élément.
4. Sinon, si *subCollection* est vide, renvoie null.
5. Sinon, retournez *subCollection* .

Pour **obtenir le ou les éléments indexés ou nommés**

**"tous"** d'une [HTMLAllCollection](#) collection nommée *nameOrIndex* :

1. Si *nameOrIndex* , [converti](#) en une valeur de chaîne JavaScript, est une [propriété d'index de tableau name](#) , renvoie le résultat de [l'obtention de l'élément indexé "all"](#) de la collection en fonction du nombre représenté par *nameOrIndex* .
2. Renvoie le résultat de [l'obtention du ou des éléments nommés "tous"](#) de la collection nommée *nameOrIndex* .

#### 2.6.2.1.1 **[[Appel]]** ( *thisArgument* , *argumentsList* )

1. Si [la taille](#) de *argumentsList* est zéro, ou si *argumentsList* [0] n'est pas défini, renvoie null.
2. Soit *nameOrIndex* le résultat de [la conversion](#) de *argumentsList* [0] en a [DOMString](#).
3. Soit *result* le résultat de [l'obtention du ou des éléments indexés ou nommés "tous"](#) à partir de ce *nameOrIndex* [HTMLAllCollection](#) donné .
4. Renvoie le résultat de [la conversion](#) de *result* en une valeur ECMAScript.

*Le thisArgument est ignoré, et donc le code tel*

*que* `Function.prototype.call.call(document.all, null, "x")` *cherchera toujours des éléments. ( document.all.call n'existe pas, car document.all n'hérite pas de Function.prototype.)*

#### 2.6.2.2 L' [HTMLFormControlsCollection](#) interface

L' [HTMLFormControlsCollection](#) interface est utilisée pour [les collections](#) d' [éléments listés](#) dans [form](#) elements.



```
[Exposed=Window]
```

```
interface HTMLFormControlsCollection : HTMLCollection {
```

```
  // inherits length and item\(\)
```

```
getter (RadioNodeList or Element)? namedItem(DOMString name);
```

```
// shadows inherited namedItem()
```

```
};
```

```
[Exposed=Window]
```

```
interface RadioNodeList : NodeList {
```

```
    attribute DOMString value;
```

```
};
```

### **collection.length**

Renvoie le nombre d'éléments dans *collection* .

```
element = collection.item(index)
```

```
element = collection[index]
```

Renvoie l'élément à l' *index* dans *collection* . Les éléments sont triés par [ordre arborescent](#) .

```
element = collection.namedItem(name)
```

✓

```
radioNodeList = collection.namedItem(name)
```

```
element = collection[name]
```

```
radioNodeList = collection[name]
```

Renvoie l'élément avec l'[ID](#) ou [name](#) le nom de la collection .

S'il existe plusieurs éléments correspondants, un [RadioNodeList](#) objet contenant tous ces éléments est renvoyé.

### **radioNodeList.value**

Renvoie la valeur du premier bouton radio coché représenté par *radioNodeList* .

```
radioNodeList.value = value
```

Vérifie le premier bouton radio représenté par *radioNodeList* qui a la valeur *value* .

Les [index de propriété pris en charge](#) par l'objet sont tels que définis pour [HTMLCollection](#) les objets.

Les [noms de propriété pris en charge](#) sont constitués des valeurs non vides de tous les attributs [id](#) et [name](#) de tous les éléments [représentés par la collection](#) , dans l'[ordre de l'arborescence](#) , en ignorant les doublons ultérieurs, avec le [id](#) d'un

élément précédant son [name](#) s'il contribue à la fois, ils diffèrent les uns des autres , et ni l'un ni l'autre n'est le double d'une entrée antérieure.

La méthode doit agir selon l'algorithme suivant : [namedItem\(name\)](#)

1. Si *name* est la chaîne vide, renvoyez null et arrêtez l'algorithme.
  2. Si, au moment où la méthode est appelée, il y a exactement un nœud dans la collection qui a soit un [id](#)attribut, soit un [name](#) attribut égal à *name* , alors retournez ce nœud et arrêtez l'algorithme.
  3. Sinon, s'il n'y a aucun nœud dans la collection qui ait un [id](#)attribut ou un [name](#)attribut égal à *name* , alors renvoyez null et arrêtez l'algorithme.
  4. Sinon, créez un nouvel [RadioNodeList](#)objet représentant une vue [en directHTMLFormControlsCollection](#) de l' objet, filtré davantage afin que les seuls nœuds de l' [RadioNodeList](#)objet soient ceux qui ont soit un [id](#)attribut, soit un [name](#)attribut égal à *name* . Les nœuds de l' [RadioNodeList](#)objet doivent être triés dans [l'ordre de l'arborescence](#) .
  5. Renvoyez cet [RadioNodeList](#)objet.
- 

Les membres de l' [RadioNodeList](#)interface hérités de l' [NodeList](#) interface doivent se comporter comme ils le feraient sur un [NodeList](#)objet.



L' [value](#)attribut IDL sur l' [RadioNodeList](#)objet, lors de l'obtention, doit renvoyer la valeur renvoyée en exécutant les étapes suivantes :

1. Soit *element* le premier élément dans [l'arborescence](#) représenté par l' [RadioNodeList](#)objet qui est un [input](#)élément dont [type](#)l'attribut est dans l' état [du bouton radio](#) et dont [la vérification](#) est vraie. Sinon, laissez-le être nul.
2. Si *l'élément* est nul, renvoie la chaîne vide.
3. Si *element* est un élément sans [value](#)attribut, renvoie la chaîne " on".
4. Sinon, renvoie la valeur de l'attribut de l' *élément*[value](#) .

Lors du paramétrage, l' [value](#)attribut IDL doit exécuter les étapes suivantes :

1. Si la nouvelle valeur est la chaîne " on" : soit *élément* le premier élément dans [l'arborescence](#) représenté par l' [RadioNodeList](#) objet qui est un [input](#) élément dont [type](#) l'attribut est dans l' état [Bouton radio](#) et dont [value](#) l'attribut de contenu est soit absent, soit présent et égal au nouveau valeur, le cas échéant. Si aucun élément de ce type n'existe, laissez à la place *element* la valeur null.

Sinon : laissez *element* être le premier élément dans [l'arborescence](#) représenté par l' [RadioNodeList](#) objet qui est un [input](#) élément dont [type](#) l'attribut est dans l' état [du bouton radio](#) et dont [value](#) l'attribut content est présent et égal à la nouvelle valeur, le cas échéant. Si aucun élément de ce type n'existe, laissez à la place *element* la valeur null.

2. Si *l'élément* n'est pas nul, définissez sa [vérification](#) sur true.

### 2.6.2.3 L' [HTMLOptionsCollection](#) interface



L' [HTMLOptionsCollection](#) interface est utilisée pour [les collections](#) d' [option](#) éléments. Il est toujours enraciné sur un [select](#) élément et possède des attributs et des méthodes qui manipulent les descendants de cet élément.

```
[Exposed=Window]
```

```
interface HTMLOptionsCollection : HTMLCollection {
```

```
  // inherits item\(\), namedItem\(\)
```

```
  [CEReactions] attribute unsigned long length; // shadows
```

```
  inherited length
```

```
  [CEReactions] setter undefined (unsigned long index,
```

```
  HTMLOptionElement? option);
```

```
  [CEReactions] undefined add((HTMLOptionElement or
```

```
  HTMLOptGroupElement) element, optional (HTMLInputElement or long)?
```

```
  before = null);
```

```
[CEReactions] undefined remove(long index);
```

```
attribute long selectedIndex;
```

```
};
```

### **`collection.length`**

Renvoie le nombre d'éléments dans *collection* .

### **`collection.length = value`**

Lorsqu'il est défini sur un nombre inférieur à la longueur existante, tronque le nombre d' [option](#) éléments dans le conteneur correspondant à *collection* .

Lorsqu'il est défini sur un nombre supérieur à la longueur existante, si ce nombre est inférieur ou égal à 100000, ajoute de nouveaux [option](#) éléments vides au conteneur correspondant à *collection* .

### **`element = collection.item(index)`**

`element = collection[index]`

Renvoie l'élément à l' *index* index dans *collection* . Les éléments sont triés par [ordre arborescent](#) .

### **`collection[index] = element`**

Lorsque *index* est un nombre supérieur au nombre d'éléments dans *collection* , ajoute de nouveaux [option](#) éléments vides dans le conteneur correspondant.

Lorsqu'il est défini sur null, supprime l'élément à l' *index* index de *collection* .

Lorsqu'il est défini sur un [option](#) élément, l'ajoute ou le remplace à l'*index* index dans *la collection* .

### **`element = collection.namedItem(name)`**

`element = collection[name]`

Renvoie l'élément avec l'[ID](#) ou [name](#) le nom de *la collection* .

S'il existe plusieurs éléments correspondants, le premier est renvoyé.

### **`collection.add(element[, before])`**

Insère *un élément* avant le nœud donné par *before* .

L' argument *avant* peut être un nombre, auquel cas *element* est inséré avant l'élément portant ce numéro, ou un élément de *collection* , auquel cas *element* est inséré avant cet élément.

Si *avant* est omis, nul ou un nombre hors plage, alors *l'élément* sera ajouté à la fin de la liste.

Lève un "[HierarchyRequestError](#)" [DOMException](#) si *l'élément* est un ancêtre de l'élément dans lequel il doit être inséré.

### **`collection.remove(index)`**

Supprime l'élément avec index *index* de *collection* .



**`collection.selectedIndex`**

Renvoie l'index du premier élément sélectionné, le cas échéant, ou -1 s'il n'y a pas d'élément sélectionné.

**`collection.selectedIndex = index`**

Remplace la sélection par l' optionélément à l'index `index` dans la collection .

Les index de propriété pris en charge par l'objet sont tels que définis pour HTMLCollection les objets.

Les lengthétapes du getter consistent à renvoyer le nombre de nœuds représentés par la collection .

Les lengthétapes du setter sont :

1. Soit *courant* le nombre de nœuds représentés par la collection .
2. Si la valeur donnée est supérieure à *current* , alors :
  1. Si la valeur donnée est supérieure à 100 000, alors retour.
  2. Soit *n* la *valeur - courant* .
  3. Ajoutez *n* nouveaux optionéléments sans attributs et sans nœuds enfants à l' selectélément sur lequel i est enraciné. Les événements de mutation doivent être déclenchés comme si un DocumentFragment contenant les nouveaux option éléments avait été inséré.
3. Si la valeur donnée est inférieure à *current* , alors :
  1. Soit *n* la *valeur courante -* .
  2. Supprimez les *n* derniers nœuds de la collection de leurs nœuds parents.

Le paramètre length ne supprime ou n'ajoute jamais optgroup d'éléments, et n'ajoute jamais de nouveaux enfants aux optgroup éléments existants (bien qu'il puisse en supprimer des enfants).

Les noms de propriété pris en charge sont constitués des valeurs non vides de tous les attributs id et name de tous les éléments représentés par la collection , dans l'ordre de l'arborescence , en ignorant les doublons ultérieurs, avec le id d'un élément précédant son name s'il contribue à la fois, ils diffèrent les uns des autres , et ni l'un ni l'autre n'est le double d'une entrée antérieure.

Lorsque l'agent utilisateur doit définir la valeur d'une nouvelle propriété indexée ou définir la valeur d'une propriété indexée existante pour un *index* de propriété donné sur une nouvelle valeur *value* , il doit exécuter l'algorithme suivant :

1. Si *value* est null, invoquez les étapes de la remove méthode avec *index* comme argument, et retournez.
2. Soit *longueur* le nombre de nœuds représentés par la collection .
3. Soit *n* l' *indice* moins *la longueur* .
4. Si *n* est supérieur à zéro, alors ajoutez a DocumentFragment composé de *n* - 1 nouveaux option éléments sans attributs et sans nœuds enfants à l' select élément sur lequel HTMLOptionsCollection est enraciné.
5. Si *n* est supérieur ou égal à zéro, ajoute la valeur à l' select élément. Sinon, remplacez l' *index* ème élément de la collection par *value* .

La méthode doit agir selon l'algorithme suivant : add(element, before)

1. Si l'élément est un ancêtre de l' select élément sur lequel HTMLOptionsCollection est enraciné, lancez un " HierarchyRequestError " DOMException .
2. Si *before* est un élément, mais que cet élément n'est pas un descendant de l' select élément sur lequel HTMLOptionsCollection est enraciné, lancez alors un " NotFoundError " DOMException .
3. Si *élément* et *avant* sont le même élément, alors retour.
4. Si *before* est un nœud, alors *référence* est ce nœud. Sinon, si *before* est un entier et qu'il y a un *avant* -ème nœud dans la collection, laissez *reference* être ce nœud. Sinon, laissez *la référence* nulle.
5. Si *reference* n'est pas null, laissez *parent* être le nœud parent de *reference* . Sinon, laissez *parent* être l' select élément sur lequel HTMLOptionsCollection est enraciné.
6. Préinsérez l'élément dans le nœud *parent* avant *la référence* .

La méthode doit agir selon l'algorithme suivant : remove(index)

1. Si le nombre de nœuds représentés par la collection est zéro, retour.
2. Si *index* n'est pas un nombre supérieur ou égal à 0 et inférieur au nombre de nœuds représentés par la collection , return.
3. Soit *element* l' *index* e élément de la collection.
4. Supprimer l'élément de son nœud parent.

L' selectedIndex attribut IDL doit agir comme l'attribut portant le même nom sur l' select élément sur lequel HTMLOptionsCollection est enraciné

### 2.6.3 L' DOMStringList interface



L' DOMStringList interface est une manière rétro non à la mode de représenter une liste de chaînes.

```
[Exposed=(Window,Worker)]  
  
interface DOMStringList {  
  
    readonly attribute unsigned long length;  
  
    getter DOMString? item(unsigned long index);  
  
    boolean contains(DOMString string);  
  
};
```

**Les nouvelles API doivent utiliser `sequence<DOMString>` ou équivalent plutôt que DOMStringList.**

**`strings.length`**

Renvoie le nombre de chaînes dans *strings* .

**`strings[index]`**

**`strings.item(index)`**

Renvoie la chaîne avec l' *index* de *strings* .

**`strings.contains(string)`**

Renvoie true si *strings* contient *string* , et false sinon.

Chaque DOMStringList objet a une [liste](#) associée .

L' DOMStringList interface [prend en charge les propriétés indexées](#) . Les [indices de propriété pris en charge](#) sont les [indices](#) de [cette](#) liste associée.

Les **length** étapes du getter consistent à renvoyer [la taille de cette](#) liste associée .

Les étapes de la méthode consistent à renvoyer l' *index* e élément dans [cette](#) liste associée, ou null si l'*index* plus un est supérieur à [la taille de cette](#) liste associée . **`item(index)`**

Les étapes de la méthode consistent à retourner true si [cette](#) liste associée [contient](#) *string* , et false sinon. **`contains(string)`**

## 2.7 Transmission sécurisée de données structurées

Pour prendre en charge le passage d'objets JavaScript, y compris [les objets de plate-forme](#) , à travers les limites [du domaine](#) , cette spécification définit l'infrastructure suivante pour la sérialisation et la désérialisation des objets, y compris dans certains cas le transfert des données sous-jacentes au lieu de les copier. Collectivement, ce processus de sérialisation/désérialisation est connu sous le nom de "clonage structuré", bien que la plupart des API effectuent des étapes de sérialisation et de désérialisation distinctes. (À l'exception notable de la [structuredClone\(\)](#) méthode.)

Cette section utilise la terminologie et les conventions typographiques de la spécification JavaScript. [\[JAVASCRIPT\]](#)

### 2.7.1 Objets sérialisables

MDN

[Les objets sérialisables](#) prennent en charge la sérialisation, puis la désérialisation, d'une manière indépendante de tout [domaine](#) donné . Cela permet de les stocker sur disque et de les restaurer ultérieurement, ou de les cloner à travers les limites [des agents](#) et même [des clusters d'agents](#) .

Tous les objets ne sont pas [des objets sérialisables](#) et tous les aspects des objets qui sont [des objets sérialisables](#) ne sont pas nécessairement conservés lorsqu'ils sont sérialisés.

[Les objets de plate-forme](#) peuvent être [des objets sérialisables](#) si leur [interface principale](#) est décorée avec l' [attribut étendu](#) `[Serializable]` IDL . Ces interfaces doivent également définir les algorithmes suivants :

**[étapes de sérialisation](#) , en prenant une valeur [d'objet de plate-forme](#) , un [enregistrement sérialisé](#) et un `forStorage` booléen**

Un ensemble d'étapes qui sérialise les données de *value* dans des champs de *serialized* . Les données résultantes sérialisées en *sérialisées* doivent être indépendantes de tout [domaine](#) .

Ces étapes peuvent lever une exception si la sérialisation n'est pas possible.

Ces étapes peuvent effectuer une [sous-sérialisation](#) pour sérialiser des structures de données imbriquées. Ils ne doivent pas appeler [StructuredSerialize](#) directement, car cela omettrait l'important argument *de mémoire* .

L'introduction de ces étapes doit omettre la mention de l'argument *forStorage* s'il n'est pas pertinent pour l'algorithme.

**étapes de désérialisation** , en prenant un [enregistrement sérialisé](#) , une [valeur d'objet de plate-forme](#) et un [domaine](#) *targetRealm*

Un ensemble d'étapes qui désérialise les données dans *serialized* , en les utilisant pour configurer la *valeur* appropriée. *value* sera une instance nouvellement créée du type [d'objet de plate-forme](#) en question, sans configuration de ses données internes ; la mise en place est le travail de ces étapes.

Ces étapes peuvent lever une exception si la désérialisation n'est pas possible.

Ces étapes peuvent effectuer une [sous-désérialisation](#) pour désérialiser des structures de données imbriquées. Ils ne doivent pas appeler directement [StructuredDeserialize](#) , car cela omettrait les arguments importants *targetRealm* et *memory* .

Il appartient à la définition des objets de plate-forme individuels de déterminer quelles données sont sérialisées et désérialisées par ces étapes. Généralement, les étapes sont très symétriques.

L' [\[Serializable\]](#) attribut étendu ne doit prendre aucun argument et ne doit apparaître que sur une interface. Il ne doit pas apparaître plus d'une fois sur une interface.

Pour un [objet de plate-forme](#) donné , seule l' [interface principale](#) de l'objet est prise en compte lors du processus de (dé)sérialisation. Ainsi, si l'héritage est impliqué dans la définition de l'interface, chaque [\[Serializable\]](#) interface annotée dans la chaîne d'héritage doit définir [des étapes de sérialisation](#) et [des étapes de désérialisation](#) autonomes , notamment en prenant en compte toutes les données importantes pouvant provenir d'interfaces héritées.

Supposons que nous définissions un objet de plate-forme *Person* auquel étaient associés deux données associées :

- une valeur de nom, qui est une chaîne ;
- et une valeur de meilleur ami, qui est soit une autre *Person* instance, soit nulle

Nous pourrions alors définir *Person* les instances comme [des objets sérialisables](#) en annotant l' *Person* interface avec l' [\[Serializable\]](#) [attribut étendu](#) et en définissant les algorithmes d'accompagnement suivants :

### [étapes de sérialisation](#)

1. Définissez *sérialisé* .[[Name]] sur la *valeur* du nom associé à la valeur.
2. Soit *serializedBestFriend* la [sous-sérialisation](#) de la *valeur* de meilleur ami associée à value .
3. Définissez *serialized* .[[BestFriend]] sur *serializedBestFriend* .

## étapes de désérialisation

1. Définissez *la valeur* du nom associé à la valeur sur *sérialisé* .[[Name]].
2. Soit *deserializedBestFriend* la sous-désérialisation de *sérialisé* .[[BestFriend]].
3. Définissez *la valeur* du meilleur ami associé à la valeur sur *deserializedBestFriend* .

Les objets définis dans la spécification JavaScript sont gérés directement par l'opération abstraite StructuredSerialize .

*À l'origine, cette spécification définissait le concept "d'objets clonables", qui pouvaient être clonés d'un domaine à un autre. Cependant, pour mieux préciser le comportement de certaines situations plus complexes, le modèle a été mis à jour pour rendre explicite la sérialisation et la désérialisation.*

### **2.7.2 Objets transférables**

Les objets transférables prennent en charge le transfert entre agents . Le transfert consiste en fait à recréer l'objet tout en partageant une référence aux données sous-jacentes, puis en détachant l'objet en cours de transfert. Ceci est utile pour transférer la propriété de ressources coûteuses. Tous les objets ne sont pas des objets transférables et tous les aspects des objets qui sont des objets transférables ne sont pas nécessairement préservés lors du transfert.

*Transférer est une opération irréversible et non idempotente. Une fois qu'un objet a été transféré, il ne peut plus être transféré ni réutilisé.*

Les objets de plate-forme peuvent être des objets transférables si leur interface principale est décorée avec l' attribut étendu **[Transferable]** IDL . Ces interfaces doivent également définir les algorithmes suivants :

**étapes de transfert** , en prenant une *valeur* d'objet de plate-forme et un Record *dataHolder*

Un ensemble d'étapes qui transfère les données en *valeur* dans les champs de *dataHolder* . Les données résultantes conservées dans *dataHolder* doivent être indépendantes de tout domaine .

Ces étapes peuvent lever une exception si le transfert n'est pas possible.

**étapes de transfert-réception** , en prenant un Record *dataHolder* et une *valeur* d'objet de plate-forme

Un ensemble d'étapes qui reçoit les données dans *dataHolder* , en l'utilisant pour configurer *la valeur* appropriée. *value* sera une instance nouvellement

créée du type [d'objet de plate-forme](#) en question, sans configuration de ses données internes ; la mise en place est le travail de ces étapes.

Ces étapes peuvent lever une exception s'il n'est pas possible de recevoir le transfert.

Il appartient à la définition des objets de plate-forme individuels de déterminer quelles données sont transférées par ces étapes. Généralement, les étapes sont très symétriques.

L' [\[Transferable\]](#) attribut étendu ne doit prendre aucun argument et ne doit apparaître que sur une interface. Il ne doit pas apparaître plus d'une fois sur une interface.

Pour un [objet de plate-forme](#) donné , seule l' [interface principale](#) de l'objet est prise en compte lors du processus de transfert. Ainsi, si l'héritage est impliqué dans la définition de l'interface, chaque [\[Transferable\]](#) interface annotée dans la chaîne d'héritage doit définir [des étapes de transfert](#) autonomes et [des étapes de réception de transfert](#) , notamment en prenant en compte toutes les données importantes pouvant provenir d'interfaces héritées.

[Les objets de plate-forme](#) qui sont [des objets transférables](#) ont un emplacement interne **[[Detached]]** . Ceci est utilisé pour s'assurer qu'une fois qu'un objet de plate-forme a été transféré, il ne peut plus être transféré.

Les objets définis dans la spécification JavaScript sont gérés directement par l'opération abstraite [StructuredSerializeWithTransfer](#) .

### 2.7.3 *StructuredSerializeInternal* ( *valeur* , *forStorage* [ , *mémoire* ] )

L' opération abstraite [StructuredSerializeInternal](#) prend en entrée une *valeur* de valeur JavaScript et la sérialise sous une forme indépendante [du domaine](#) , représentée ici par un [Record](#) . Ce formulaire sérialisé contient toutes les informations nécessaires pour se désérialiser ultérieurement en une nouvelle valeur JavaScript dans un domaine différent.

Ce processus peut lever une exception, par exemple lors d'une tentative de sérialisation d'objets non sérialisables.

1. Si *la mémoire* n'a pas été fournie, laissez *la mémoire* être une [carte](#) vide .

*Le but de la carte mémoire est d'éviter de sérialiser deux fois les objets. Cela finit par préserver les cycles et l'identité des objets dupliqués dans les graphes.*

2. Si *la mémoire* [ *valeur* ] [existe](#) , alors retournez *la mémoire* [ *valeur* ] .



3. Que *deep* soit faux.
4. Si [Type](#) ( *value* ) est Undefined, Null, Boolean, Number, BigInt ou String, alors retournez { [[Type]] : "primitive", [[Value]] : *value* }.
5. Si [Type](#) ( *value* ) est Symbol, lancez un ["DataCloneError" DOMException](#) .
6. Soit *sérialisé* une valeur non initialisée.
7. Si *la valeur* a un emplacement interne [[BooleanData]], définissez *sérialisé* sur { [[Type]] : "Boolean", [[BooleanData]] : *value* .[[BooleanData]] }.
8. Sinon, si *la valeur* a un emplacement interne [[NumberData]], définissez *sérialisé* sur { [[Type]] : "Number", [[NumberData]] : *value* .[[NumberData]] }.
9. Sinon, si *la valeur* a un emplacement interne [[BigIntData]], définissez *sérialisé* sur { [[Type]] : "BigInt", [[BigIntData]] : *valeur* .[[BigIntData]] }.
10. Sinon, si *la valeur* a un emplacement interne [[StringData]], définissez *sérialisé* sur { [[Type]] : "String", [[StringData]] : *valeur* .[[StringData]] }.
11. Sinon, si *la valeur* a un emplacement interne [[DateValue]], définissez *sérialisé* sur { [[Type]] : "Date", [[DateValue]] : *valeur* .[[DateValue]] }.
12. Sinon, si *la valeur* a un emplacement interne [[RegExpMatcher]], définissez *sérialisé* sur { [[Type]] : "RegExp", [[RegExpMatcher]] : *valeur* .[[RegExpMatcher]], [[OriginalSource]] : *valeur* .[[OriginalSource]], [[OriginalFlags]] : *valeur* .[[OriginalFlags]] }.
13. Sinon, si *la valeur* a un emplacement interne [[ArrayBufferData]], alors :
  1. Si [IsSharedArrayBuffer](#) ( *valeur* ) est vrai, alors :
    1. Si la [capacité d'isolement d'origine croisée](#) de [l'objet de paramètres actuel](#) est fausse, lancez un [" " .DataCloneError DOMException](#)

*Cette vérification n'est nécessaire que lors de la sérialisation (et non lors de la désérialisation) car la [capacité d'isolement d'origine croisée](#) ne peut pas changer dans le temps et ne [SharedArrayBuffer](#) peut pas quitter un [cluster d'agents](#) .*

  - 2. Si *forStorage* est vrai, lancez un ["DataCloneError" DOMException](#) .
  - 3. Si *la valeur* a un emplacement interne [[ArrayBufferMaxByteLength]], définissez *sérialisé* sur { [[Type]] :



"GrowableSharedArrayBuffer",  
 [[ArrayBufferData]] : *valeur* .[[ArrayBufferData]],  
 [[ArrayBufferByteLengthData]] : *valeur* .[  
 ArrayBufferByteLengthData]],  
 [[ArrayBufferMaxByteLength]] : *valeur* .[[ArrayBufferMaxByteLen  
 gth]], [[AgentCluster]] : [cluster d'agents](#) de [l'agent environnant](#) }.

4. Sinon, définissez *sérialisé* sur { [[Type]] : "SharedArrayBuffer",  
 [[ArrayBufferData]] : *valeur* .[[ArrayBufferData]],  
 [[ArrayBufferByteLength]] : *valeur* .[[ArrayBufferByteLength]],  
 [[AgentCluster]] : le [cluster d'agents](#) de [l'agent environnant](#) }.

2. Sinon:

1. Si [IsDetachedBuffer](#) ( *value* ) est vrai, lancez  
 un "[DataCloneError](#)" [DOMException](#) .
2. Soit *size* la *valeur* .[[ArrayBufferByteLength]].
3. Soit *dataCopy* ? [CreateByteDataBlock](#) ( *taille* ).

*Cela peut lever une [RangeError](#) exception en cas d'échec d'allocation.*

4. Effectuez [CopyDataBlockBytes](#) ( *dataCopy* ,  
 0, *value* .[[ArrayBufferData]], 0, *size* ).
5. Si *la valeur* a un emplacement interne  
 [[ArrayBufferMaxByteLength]], définissez *sérialisé* sur { [[Type]] :  
 "ResizableArrayBuffer", [[ArrayBufferData]] : *dataCopy* ,  
 [[ArrayBufferByteLength]] : *size* ,  
 [[ArrayBufferMaxByteLength]] : *valeur* .[[ArrayBufferMaxByteLen  
 gth]] }.
6. Sinon, définissez *sérialisé* sur { [[Type]]: "ArrayBuffer",  
 [[ArrayBufferData]]: *dataCopy* , [[ArrayBufferByteLength]]: *size* }.

14. Sinon, si *la valeur* a un emplacement interne [[ViewedArrayBuffer]], alors :

1. Si [IsArrayBufferViewOutOfBounds](#) ( *value* ) est vrai, lancez  
 un "[DataCloneError](#)" [DOMException](#) .
2. Soit *buffer* la valeur de l'emplacement interne [[ViewedArrayBuffer]]  
 de *value* .
3. Soit *bufferSerialized* être  
 ? [StructuredSerializeInternal](#) ( *buffer* , *forStorage* , *memory* ).
4. [Assert](#) : *bufferSerialized* .[[Type]] est "ArrayBuffer",  
 "ResizableArrayBuffer", "SharedArrayBuffer" ou  
 "GrowableSharedArrayBuffer".

5. Si *la valeur* a un emplacement interne `[[DataView]]`,  
définissez *sérialisé* sur `{ [[Type]] : "ArrayBufferView", [[Constructor]] : "DataView", [[ArrayBufferSerialized]] : bufferSerialized, [[ByteLength]] : valeur .[[ByteLength]], [[ByteOffset]] : valeur .[[ByteOffset]] }`.
6. Sinon:
  1. [Assert](#) : *la valeur* a un emplacement interne `[[TypedArrayName]]`.
  2. Définissez *sérialisé* sur `{ [[Type]] : "ArrayBufferView", [[Constructor]] : valeur .[[TypedArrayName]], [[ArrayBufferSerialized]] : bufferSerialized, [[ByteLength]] : valeur .[[ByteLength]], [[ByteOffset]] : valeur .[[ByteOffset]], [[ArrayLength]] : valeur .[[ArrayLength]] }`.

15. Sinon, si *la valeur* a un emplacement interne `[[MapData]]`, alors :

1. Définissez *sérialisé* sur `{ [[Type]] : "Map", [[MapData]] : une nouvelle liste vide }`.
2. Définissez *profondeur* sur vrai.

16. Sinon, si *la valeur* a un emplacement interne `[[SetData]]`, alors :

1. Définissez *sérialisé* sur `{ [[Type]] : "Set", [[SetData]] : une nouvelle liste vide }`.
2. Définissez *profondeur* sur vrai.

17. Sinon, si *la valeur* a un emplacement interne `[[ErrorData]]` et que *la valeur* n'est pas un [objet de plate-forme](#) , alors :

1. Soit *le nom* ? [Obtenir](#) ( *valeur* , "nom").
2. Si *le nom* n'est pas l'un des "Error", "EvalError", "RangeError", "ReferenceError", "SyntaxError", "TypeError" ou "URIError", alors définissez le nom sur "Error " .
3. Soit *valueMessageDesc* être ? *valeur* .[[GetOwnProperty]](" *message* ").
4. Laissez *message* être indéfini  
si [IsDataDescriptor](#) ( *valueMessageDesc* ) est faux, et  
? [ToString](#) ( *valueMessageDesc* .[[Value]]) sinon.
5. Définissez *sérialisé* sur `{ [[Type]] : "Error", [[Name]] : name , [[Message]] : message }`.
6. Les agents utilisateurs doivent joindre une représentation sérialisée de toutes les données d'accompagnement intéressantes qui ne sont pas encore spécifiées, notamment la `stack` propriété, à *serialized* .

Voir la proposition Error Stacks pour les travaux en cours sur la spécification de ces données. [\[JSERRORSTACKS\]](#)

18. Sinon, si *value* est un objet exotique Array, alors :

1. Soit *valueLenDescriptor* être ? [OrdinaryGetOwnProperty](#) ( *valeur* , "length").
2. Soit *valueLen* être *valueLenDescriptor* .[[Value]].
3. Définissez *sérialisé* sur { [[Type]] : "Array", [[Length]] : *valueLen* , [[Properties]] : une nouvelle [liste](#) vide }.
4. Définissez *profondeur* sur vrai.

19. Sinon, si *valeur* est un [objet plateforme](#) qui est un [objet sérialisable](#) :

1. Si *la valeur* a un emplacement interne [\[\[Detached\]\]](#) dont la valeur est true, lancez un ["DataCloneError"](#) [DOMException](#) .
2. Soit *typeString* l'identifiant de l' [interface principale](#) de *value* .
3. Définissez *sérialisé* sur { [[Type]] : *typeString* }.
4. Définissez *profondeur* sur vrai.

20. Sinon, si *value* est un [objet platform](#) , lancez un ["DataCloneError"](#) [DOMException](#) .

21. Sinon, si [IsCallable](#) ( *value* ) est true, lancez un ["DataCloneError"](#) [DOMException](#) .

22. Sinon, si *la valeur* a un emplacement interne autre que [\[\[Prototype\]\]](#) ou [\[\[Extensible\]\]](#), lancez un ["DataCloneError"](#) [DOMException](#) .

Par exemple, un emplacement interne [\[\[PromiseState\]\]](#) ou [\[\[WeakMapData\]\]](#).

23. Sinon, si *value* est un objet exotique et que *value* n'est pas l' objet intrinsèque [%Object.prototype%](#) associé à un [domaine](#) , lancez un ["DataCloneError"](#) [DOMException](#) .

Par exemple, un objet proxy.

24. Sinon:

1. Définissez *sérialisé* sur { [[Type]] : "Objet", [[Propriétés]] : une nouvelle [liste](#) vide }.
2. Définissez *profondeur* sur vrai.

[%Object.prototype%](#) finira par être géré via cette étape et les étapes suivantes. Le résultat final est que son caractère exotique est ignoré, et après

la désérialisation, le résultat sera un objet vide (pas un objet exotique prototype immuable ).

25. Définissez la mémoire [ valeur ] sur sérialisé .

26. Si *deep* est vrai, alors :

1. Si la valeur a un emplacement interne [[MapData]], alors :

1. Soit *copiedList* une nouvelle List vide .

2. Pour chaque Record { [[Key]], [[Value]]  
} entrée de *value* .[[MapData]] :

1. Soit *copiedEntry* un nouvel enregistrement {  
[[Key]]: *entry* .[[Key]], [[Value]]: *entry* .[[Value]] }.

2. Si *copiedEntry* .[[Key]] n'est pas la valeur  
spéciale *empty* , ajoutez *copiedEntry* à *copiedList* .

3. Pour chaque entrée Record { [[Key]], [[Value]] } de *copiedList* :

1. Soit *serializedKey* ? StructuredSerializeInternal ( entrée .[[  
Key]], *forStorage* , *memory* ).

2. Soit *serializedValue* ? StructuredSerializeInternal ( entrée  
.[[Valeur]], *forStorage* , *memory* ).

3. Ajoutez { [[Key]] : *serializedKey* ,  
[[Value]] : *serializedValue* } à *sérialisé* .[[MapData]].

2. Sinon, si la valeur a un emplacement interne [[SetData]], alors :

1. Soit *copiedList* une nouvelle List vide .

2. Pour chaque entrée de *value* .[[SetData]] :

1. Si l'entrée n'est pas la valeur  
spéciale *empty* , ajoute l'entrée à *copiedList* .

3. Pour chaque entrée de *copiedList* :

1. Soit *serializedEntry* ? StructuredSerializeInternal ( entrée ,  
*forStorage* , *mémoire* ).

2. Ajouter *serializedEntry* au *sérialisé* .[[SetData]].

3. Sinon, si *value* est un objet de plate-forme qui est un objet sérialisable ,  
effectuez les étapes de sérialisation pour l'interface  
principale de *value* , *value* , *serialized* et *forStorage* .

Les étapes de sérialisation peuvent nécessiter d'effectuer une **sous-sérialisation** . Il s'agit d'une opération qui prend en entrée une

valeur *subValue* et renvoie [StructuredSerializeInternal](#) ( *subValue* , *forStorage* , *memory* ). (En d'autres termes, une [sous-sérialisation](#) est une spécialisation de [StructuredSerializeInternal](#) pour être cohérent dans cet appel.)

4. Sinon, pour chaque *clé* dans ! [EnumerableOwnProperties](#) ( *valeur* , *clé* ):
  1. Si ! [HasOwnProperty](#) ( *value* , *key* ) est vrai, alors :
    1. Soit *inputValue* être ? *valeur* .[[Get]]( *clé* , *valeur* ).
    2. Soit *outputValue* être ? [StructuredSerializeInternal](#) ( *inputValue* , *forStorage* , *memory* ).
    3. [Ajoutez](#) { [[Key]]: *key* , [[Value]]: *outputValue* } aux .[[Properties]] *sérialisés* .

## 27. Retour *sérialisé* .

Il est important de réaliser que les [enregistrements](#) produits par [StructuredSerializeInternal](#) peuvent contenir des "pointeurs" vers d'autres enregistrements qui créent des références circulaires. Par exemple, lorsque nous passons l'objet JavaScript suivant dans [StructuredSerializeInternal](#) :

```
const o = {};  
o.myself = o;
```

il produit le résultat suivant :

```
{  
  [[Type]] : "Objet",  
  [[Propriétés]] : "  
    {  
      [[Key]] : "moi-même",  
      [[Valeur]] : <un pointeur vers toute cette structure>  
    }  
  »  
}
```

### 2.7.4 *StructuredSerialize* ( *valeur* )

1. Retour ? [StructuredSerializeInternal](#) ( *valeur* , faux ).

### 2.7.5 *StructuredSerializeForStorage* ( *valeur* )

1. Retour ? [StructuredSerializeInternal](#) ( *valeur* , vrai).

### 2.7.6 *StructuredDeserialize* ( *sérialisé* , *targetRealm* [ , *mémoire* ] )

L'opération abstraite [StructuredDeserialize](#) prend en entrée un [Record](#) *serialized* , qui a été précédemment produit par [StructuredSerialize](#) ou [StructuredSerializeForStorage](#) , et le déséréalise en une nouvelle valeur JavaScript, créée dans *targetRealm* .

Ce processus peut lever une exception, par exemple lors de la tentative d'allocation de mémoire pour les nouveaux objets (en particulier `ArrayBuffer` les objets).

1. Si *la mémoire* n'a pas été fournie, laissez *la mémoire* être une [carte](#) vide .

*Le but de la carte mémoire est d'éviter de déséréaliser les objets deux fois. Cela finit par préserver les cycles et l'identité des objets dupliqués dans les graphes.*

2. Si *la mémoire* [ *sérialisée* ] [existe](#) , alors retournez *la mémoire* [ *sérialisée* ] .
3. Que *deep* soit faux.
4. Soit *value* une valeur non initialisée.
5. Si *sérialisé* .[[Type]] est "primitif", alors définissez *la valeur* sur *sérialisé* .[[Valeur]].
6. Sinon, si *sérialisé* .[[Type]] est "Booléen", alors définissez *la valeur* sur un nouvel objet booléen dans *targetRealm* dont la valeur d'emplacement interne [[BooleanData]] est *sérialisée* .[[BooleanData]].
7. Sinon, si *sérialisé* .[[Type]] est "Number", alors définissez *la valeur* sur un nouvel objet Number dans *targetRealm* dont la valeur d'emplacement interne [[NumberData]] est *sérialisée* .[[NumberData]].
8. Sinon, si *sérialisé* .[[Type]] est "BigInt", alors définissez *la valeur* sur un nouvel objet BigInt dans *targetRealm* dont la valeur d'emplacement interne [[BigIntData]] est *sérialisée* .[[BigIntData]].
9. Sinon, si *sérialisé* .[[Type]] est "String", alors définissez *la valeur* sur un nouvel objet String dans *targetRealm* dont la valeur d'emplacement interne [[StringData]] est *sérialisée* .[[StringData]].
10. Sinon, si *sérialisé* .[[Type]] est "Date", alors définissez *la valeur* sur un nouvel objet Date dans *targetRealm* dont la [[DateValue]] valeur d'emplacement interne est *sérialisée* .[[DateValue]].

11. Sinon, si *sérialisé* .[[Type]] est "RegExp", alors définissez *la valeur* sur un nouvel objet RegExp dans *targetRealm* dont la valeur d'emplacement interne [[RegExpMatcher]] est *sérialisée* .[[RegExpMatcher]], dont [[OriginalSource]] interne la valeur d'emplacement est *sérialisée* .[[OriginalSource]], et dont la valeur d'emplacement interne [[OriginalFlags]] est *sérialisée* .[[OriginalFlags]].

12. Sinon, si *sérialisé* .[[Type]] est "SharedArrayBuffer", alors :

1. Si [le cluster d'agents](#) correspondant de *targetRealm* n'est pas *sérialisé* .[[AgentCluster]], lancez alors un `" ".DataCloneError DOMException`
2. Sinon, définissez *la valeur* sur un nouvel objet SharedArrayBuffer dans *targetRealm* dont la valeur d'emplacement interne [[ArrayBufferData]] est *sérialisée* .[[ArrayBufferData]] et dont la valeur d'emplacement interne [[ArrayBufferByteLength]] est *sérialisée* .[[ArrayBufferByteLength]].

13. Sinon, si *sérialisé* .[[Type]] est "GrowableSharedArrayBuffer", alors :

1. Si [le cluster d'agents](#) correspondant de *targetRealm* n'est pas *sérialisé* .[[AgentCluster]], lancez alors un `" ".DataCloneError DOMException`
2. Sinon, définissez *la valeur* sur un nouvel objet SharedArrayBuffer dans *targetRealm* dont la valeur d'emplacement interne [[ArrayBufferData]] est *sérialisée* .[[ArrayBufferData]], dont la valeur d'emplacement interne [[ArrayBufferByteLengthData]] est *sérialisée* .[[ArrayBufferByteLengthData]], et dont [ [[ArrayBufferMaxByteLength]] la valeur de l'emplacement interne est *sérialisée* .[[ArrayBufferMaxByteLength]].

14. Sinon, si *sérialisé* .[[Type]] est "ArrayBuffer", alors définissez *la valeur* sur un nouvel objet ArrayBuffer dans *targetRealm* dont la valeur d'emplacement interne [[ArrayBufferData]] est *sérialisée* .[[ArrayBufferData]], et dont [[ArrayBufferByteLength]] la valeur de l'emplacement interne est *sérialisée* .[[ArrayBufferByteLength]].

Si cela lève une exception, attrapez-la, puis lancez un `"DataCloneError" DOMException` .

*Cette étape peut lever une exception s'il n'y a pas assez de mémoire disponible pour créer un tel objet ArrayBuffer.*

15. Sinon, si *sérialisé* .[[Type]] est "ResizableArrayBuffer", alors définissez *la valeur* sur un nouvel objet ArrayBuffer dans *targetRealm* dont la valeur d'emplacement interne [[ArrayBufferData]] est *sérialisée* .[[ArrayBufferData]], dont [[ArrayBufferByteLength]] interne La valeur d'emplacement est *sérialisée* .[[ArrayBufferByteLength]], et dont la valeur d'emplacement interne [[ArrayBufferMaxByteLength]] est un .[[ArrayBufferMaxByteLength]] *sérialisé* .



Si cela lève une exception, attrapez-la, puis lancez un `"DataCloneError"` `DOMException` .

*Cette étape peut lever une exception s'il n'y a pas assez de mémoire disponible pour créer un tel objet `ArrayBuffer`.*

16. Sinon, si `sérialisé` `[[Type]]` est `"ArrayBufferView"`, alors :

1. Soit `deserializedArrayBuffer` ? [StructuredDeserialize](#) ( `sérialisé` `[[ArrayBufferSerialized]]`, `targetRealm` , `memory` ).
2. Si `sérialisé` `[[Constructor]]` est `"DataView"`, alors définissez *la valeur* sur un nouvel objet `DataView` dans `targetRealm` dont la valeur d'emplacement interne `[[ViewedArrayBuffer]]` est `deserializedArrayBuffer` , dont la valeur d'emplacement interne `[[ByteLength]]` est `sérialisée` `[[ByteLength]]`, et dont la valeur d'emplacement interne `[[ByteOffset]]` est `sérialisée` `[[ByteOffset]]`.
3. Sinon, définissez *la valeur* sur un nouvel objet tableau typé dans `targetRealm` , en utilisant le constructeur donné par `serialized` `[[Constructor]]`, dont la valeur d'emplacement interne `[[ViewedArrayBuffer]]` est `deserializedArrayBuffer` , dont la valeur d'emplacement interne `[[TypedArrayName]]` est `serialized` `[[Constructor]]`, dont la valeur d'emplacement interne `[[ByteLength]]` est `sérialisée` `[[ByteLength]]`, dont la valeur d'emplacement interne `[[ByteOffset]]` est `sérialisée` `[[ByteOffset]]`, et dont la valeur `[[ArrayLength]]` interne la valeur de l'emplacement est `sérialisée` `[[ArrayLength]]`.

17. Sinon, si `sérialisé` `[[Type]]` est `"Carte"`, alors :

1. Définissez *la valeur* sur un nouvel objet `Map` dans `targetRealm` dont la valeur d'emplacement interne `[[MapData]]` est une nouvelle [List](#) vide .
2. Définissez *profondeur* sur vrai.

18. Sinon, si `sérialisé` `[[Type]]` est `"Set"`, alors :

1. Définissez *la valeur* sur un nouvel objet `Set` dans `targetRealm` dont la valeur d'emplacement interne `[[SetData]]` est une nouvelle [List](#) vide .
2. Définissez *profondeur* sur vrai.

19. Sinon, si `sérialisé` `[[Type]]` est `"Array"`, alors :

1. Laissez `outputProto` être `targetRealm` `[[Intrinsics]].[[ %Array.prototype % ]]`.
2. Réglez *la valeur* sur ! [ArrayCreate](#) ( `sérialisé` `[[Length]]`, `outputProto` ).
3. Définissez *profondeur* sur vrai.



20. Sinon, si *sérialisé* .[[Type]] est "Objet", alors :

1. Définissez *la valeur* sur un nouvel objet dans *targetRealm* .
2. Définissez *profondeur* sur vrai.

21. Sinon, si *sérialisé* .[[Type]] est "Erreur", alors :

1. Soit *prototype* soit [%Error.prototype%](#) .
2. Si *sérialisé* .[[Name]] est "EvalError", alors définissez *prototype* sur [%EvalError.prototype%](#) .
3. Si *sérialisé* .[[Name]] est "RangeError", alors définissez *prototype* sur [%RangeError.prototype%](#) .
4. Si *sérialisé* .[[Name]] est "ReferenceError", alors définissez *prototype* sur [%ReferenceError.prototype%](#) .
5. Si *sérialisé* .[[Name]] est "SyntaxError", alors définissez *prototype* sur [%SyntaxError.prototype%](#) .
6. Si *sérialisé* .[[Name]] est "TypeError", alors définissez *prototype* sur [%TypeError.prototype%](#) .
7. Si *sérialisé* .[[Name]] est "URIError", alors définissez *prototype* sur [%URIError.prototype%](#) .
8. Laissez *le message* être *sérialisé* .[[Message]].
9. Définissez *la valeur* sur [OrdinaryObjectCreate](#) ( *prototype* , « [[ErrorData]] » ).
10. Soit *messageDesc* être [PropertyDescriptor](#) { [[Valeur]] : *message* , [[Writable]] : vrai, [[Enumerable]] : faux, [[Configurable]] : vrai }.
11. Si *le message* n'est pas indéfini, alors exécutez  
! [OrdinaryDefineOwnProperty](#) ( *valeur* , " *message* " , *messageDesc* ).
12. Toute donnée d'accompagnement intéressante attachée à *sérialisé* doit être désérialisée et attachée à *value* .

22. Sinon:

1. Laissez *interfaceName* être *sérialisé* .[[Type]].
2. Si l'interface identifiée par *interfaceName* n'est pas [exposée](#) dans *targetRealm* , lancez un "[DataCloneError](#)" [DOMException](#) .
3. Définissez *la valeur* sur une nouvelle instance de l'interface identifiée par *interfaceName* , créée dans *targetRealm* .

4. Définissez *profondeur* sur vrai.

23. Définissez la mémoire [ *sérialisée* ] sur la valeur .

24. Si *deep* est vrai, alors :

1. Si *sérialisé* .[[Type]] est "Carte", alors :

1. Pour chaque entrée Record { [[Key]], [[Value]]  
} de .[[MapData]] *sérialisé* :

1. Soit *deserializedKey* ? StructuredDeserialize ( entrée .[[Key]], *targetRealm* , *memory* ).

2. Soit *deserializedValue* être ? StructuredDeserialize ( entrée .[[Value]], *targetRealm* , *memory* ).

3. Ajoutez { [[Key]] : *deserializedKey* ,  
[[Value]] : *deserializedValue* } à la valeur .[[MapData]].

2. Sinon, si *sérialisé* .[[Type]] est "Set", alors :

1. Pour chaque entrée de .[[SetData]] *sérialisé* :

1. Soit *deserializedEntry* ? StructuredDeserialize ( *entry* , *targetRealm* , *memory* ).

2. Ajouter *deserializedEntry* à la valeur .[[SetData]].

3. Sinon, si *sérialisé* .[[Type]] est "Array" ou "Object", alors :

1. Pour chaque entrée Record { / [[Key]], [[Value]]  
} de *sérialisé* .[[Properties]] :

1. Soit *deserializedValue* être ? StructuredDeserialize ( entrée .[[Value]], *targetRealm* , *memory* ).

2. Que le résultat soit  
! CreateDataProperty ( *value* , *entry* .[[Key]], *deserializedValue* ).

3. Assert : le résultat est vrai.

4. Sinon:

1. Effectuez les étapes de désérialisation appropriées pour  
l'interface identifiée par *serialized* .[[Type]], étant  
donné *serialized* , *value* et *targetRealm* .

Les étapes de désérialisation peuvent nécessiter d'effectuer  
une **sous-désérialisation** . Il s'agit d'une opération qui prend en  
entrée un Record *subSerialized* précédemment *sérialisé* et  
renvoie StructuredDeserialize ( *subSerialized* , *targetRealm* , *me*

mory ). (En d'autres termes, une [sous-désérialisation](#) est une spécialisation de [StructuredDeserialize](#) pour être cohérent dans cet appel.)

25. Valeur de retour .

### 2.7.7 *StructuredSerializeWithTransfer* ( *value* , *transferList* )

1. Soit la mémoire une [carte](#) vide .

*En plus de la façon dont il est utilisé normalement par [StructuredSerializeInternal](#) , dans cet algorithme, la mémoire est également utilisée pour garantir que [StructuredSerializeInternal](#) ignore les éléments de transferList , et nous laissons faire notre propre gestion à la place.*

2. [Pour chaque](#) transférable de *transferList* :

1. Si *transférable* n'a ni emplacement interne `[[ArrayBufferData]]` ni emplacement interne `[[Detached]]` , lancez alors un `"DataCloneError"` `DOMException` .
2. Si *transférable* a un emplacement interne `[[ArrayBufferData]]` et [que](#) `IsSharedArrayBuffer` ( *transférable* ) est vrai, lancez un `"DataCloneError"` `DOMException` .
3. Si la mémoire [ *transférable* ] [existe](#) , lancez un `"DataCloneError"` `DOMException` .
4. [Définissez](#) la mémoire [ *transférable* ] sur { `[[Type]]` } : une valeur non initialisée }.

*transférable n'est pas encore transféré car le transfert a des effets secondaires et [StructuredSerializeInternal](#) doit être en mesure de lancer en premier.*

3. Soit *sérialisé* ? [StructuredSerializeInternal](#) ( *valeur* , faux, *mémoire* ).

4. Soit *transferDataHolders* un nouveau [List](#) vide .

5. [Pour chaque](#) transférable de *transferList* :

1. Si *transférable* a un emplacement interne `[[ArrayBufferData]]` et `IsDetachedBuffer` ( *transférable* ) est vrai, lancez un `"DataCloneError"` `DOMException` .

2. Si *transférable* a un emplacement interne [\[\[Detached\]\]](#) et *transférable* . [\[\[Detached\]\]](#) est vrai, puis lancez un ["DataCloneError"](#) [DOMException](#) .
3. Soit *dataHolder* la mémoire [ *transférable* ].
4. Si *transférable* a un emplacement interne [\[\[ArrayBufferData\]\]](#), alors :
  1. Si *transférable* a un emplacement interne [\[\[ArrayBufferMaxByteLength\]\]](#), alors :
    1. Définissez *dataHolder* .[\[\[Type\]\]](#) sur "ResizableArrayBuffer".
    2. Définissez *dataHolder* .[\[\[ArrayBufferData\]\]](#) sur *transférable* .[\[\[ArrayBufferData\]\]](#).
    3. Définissez *dataHolder* .[\[\[ArrayBufferByteLength\]\]](#) sur *transférable* .[\[\[ArrayBufferByteLength\]\]](#).
    4. Définissez *dataHolder* .[\[\[ArrayBufferMaxByteLength\]\]](#) sur *transférable* .[\[\[ArrayBufferMaxByteLength\]\]](#).
  2. Sinon:
    1. Définissez *dataHolder* .[\[\[Type\]\]](#) sur "ArrayBuffer".
    2. Définissez *dataHolder* .[\[\[ArrayBufferData\]\]](#) sur *transférable* .[\[\[ArrayBufferData\]\]](#).
    3. Définissez *dataHolder* .[\[\[ArrayBufferByteLength\]\]](#) sur *transférable* .[\[\[ArrayBufferByteLength\]\]](#).
3. Effectuer ? [DetachArrayBuffer](#) ( *transférable* ) .

Les spécifications peuvent utiliser le slot interne [\[\[ArrayBufferDetachKey\]\]](#) pour empêcher [ArrayBuffer](#) le détachement des s. Ceci est utilisé dans WebAssembly JavaScript Interface , par exemple. [\[WASMJS\]](#)

5. Sinon:
  1. [Assert](#) : *transférable* est un [objet plate-forme](#) qui est un [objet transférable](#) .
  2. Soit *interfaceName* l'identifiant de l' [interface principale](#) de *transférable* .
  3. Définissez *dataHolder* .[\[\[Type\]\]](#) sur *interfaceName* .

4. [Effectuez les étapes de transfert](#) appropriées pour l'interface identifiée par *interfaceName* , étant donné *transférable* et *dataHolder* .
5. Ensemble *transférable* . [\[\[Détaché\]\]](#) à vrai.
6. [Ajoutez](#) *dataHolder* à *transferDataHolders* .
6. Renvoie { [\[\[Serialized\]\]](#) : *sérialisé* , [\[\[TransferDataHolders\]\]](#) : *transferDataHolders* }.

### 2.7.8 *StructuredDeserializeWithTransfer* ( *serializeWithTransferResult* , *targetRealm* )

1. Soit *la mémoire* une [carte](#) vide .

*Analogue à [StructuredSerializeWithTransfer](#) , en plus de la façon dont il est utilisé normalement par [StructuredDeserialize](#) , dans cet algorithme, la mémoire est également utilisée pour s'assurer que [StructuredDeserialize](#) ignore les éléments dans *serializeWithTransferResult* .[\[\[TransferDataHolders\]\]](#), et laissez-nous faire notre propre gestion à la place.*

2. Soit *transferValues* un nouveau [List](#) vide .
3. [Pour](#) [chaque](#) *transferDataHolder* de *serializeWithTransferResult* .[\[\[TransferDataHolders\]\]](#) :
  1. Soit *value* une valeur non initialisée.
  2. Si *transferDataHolder* .[\[\[Type\]\]](#) est "ArrayBuffer", alors définissez *la valeur* sur un nouvel objet ArrayBuffer dans *targetRealm* dont la valeur d'emplacement interne [\[\[ArrayBufferData\]\]](#) est *transferDataHolder* .[\[\[ArrayBufferData\]\]](#), et dont l'emplacement interne [\[\[ArrayBufferByteLength\]\]](#) la valeur est *transferDataHolder* .[\[\[ArrayBufferByteLength\]\]](#).

*Dans les cas où la mémoire d'origine occupée par [\[\[ArrayBufferData\]\]](#) est accessible lors de la désérialisation, il est peu probable que cette étape lève une exception, car aucune nouvelle mémoire n'a besoin d'être allouée : la mémoire occupée par [\[\[ArrayBufferData\]\]](#) est simplement transféré dans le nouveau ArrayBuffer. Cela peut être vrai, par exemple, lorsque les domaines source et cible sont dans le même processus.*

3. Sinon, si *transferDataHolder* .[[Type]] est "ResizableArrayBuffer", alors définissez *la valeur* sur un nouvel objet *ArrayBuffer* dans *targetRealm* dont la valeur d'emplacement interne [[*ArrayBufferData*]] est *transferDataHolder* .[[*ArrayBufferData*]], dont [[*ArrayBufferByteLength*]] interne La valeur d'emplacement est *transferDataHolder* .[[*ArrayBufferByteLength*]], et dont la valeur d'emplacement interne [[*ArrayBufferMaxByteLength*]] est *transferDataHolder* .[[*ArrayBufferMaxByteLength*]].

*Pour la même raison que l'étape précédente, il est également peu probable que cette étape lève une exception.*

4. Sinon:
  1. Soit *interfaceName* être *transferDataHolder* .[[Type]].
  2. Si l'interface identifiée par *interfaceName* n'est pas exposée dans *targetRealm* , lancez un "[DataCloneError](#)" [DOMException](#) .
  3. Définissez *la valeur* sur une nouvelle instance de l'interface identifiée par *interfaceName* , créée dans *targetRealm* .
  4. [Effectuez les étapes de réception de transfert](#) appropriées pour l'interface identifiée par *interfaceName* avec *transferDataHolder* et *value* .
5. [Définissez](#) la mémoire [ *transferDataHolder* ] sur la valeur .
6. [Ajouter](#) une valeur à *transferValues* .
4. Soit désérialisé ? [StructuredDeserialize](#) ( *serializeWithTransferResult* .[[Serialized]], *targetRealm* , *memory* ).
5. Renvoi { [[Désérialisé]] : *désérialisé* , [[ValeursTransférées]] : *valeurs transférées* }.

### 2.7.9 Exécution de la sérialisation et transfert à partir d'autres spécifications

D'autres spécifications peuvent utiliser les opérations abstraites définies ici. Ce qui suit fournit des indications sur le moment où chaque opération abstraite est généralement utile, avec des exemples.

[StructuredSerializeWithTransfer](#)  
[StructuredDeserializeWithTransfer](#)

Cloner une valeur vers un autre [domaine](#) , avec une liste de transfert, mais où le domaine cible n'est pas connu à l'avance. Dans ce cas, l'étape de sérialisation peut être effectuée immédiatement, l'étape de désérialisation étant retardée jusqu'à ce que le domaine cible soit connu.

`messagePort.postMessage()` utilise cette paire d'opérations abstraites, car le domaine de destination n'est pas connu tant que le [MessagePort](#) n'a pas été expédié .

## [StructuredSerialize](#)

### [StructuredSerializeForStorage](#)

#### [Désérialiser structuré](#)

Création d'un instantané indépendant [du domaine](#) d'une valeur donnée qui peut être enregistrée pendant une durée indéfinie, puis réifiée ultérieurement dans une valeur JavaScript, éventuellement plusieurs fois.

[StructuredSerializeForStorage](#) peut être utilisé dans les situations où la sérialisation est prévue pour être stockée de manière persistante, au lieu d'être transmise entre les domaines. Il se lance lors d'une tentative de sérialisation [SharedArrayBuffer](#) d'objets, car le stockage de la mémoire partagée n'a pas de sens. De même, il peut lancer ou éventuellement avoir un comportement différent lorsqu'il reçoit un [objet de plate-forme](#) avec [des étapes de sérialisation](#) personnalisées lorsque l'argument *forStorage* est vrai.

`history.pushState()` et `history.replaceState()` utilisent [StructuredSerializeForStorage](#) sur les objets d'état fournis par l'auteur, en les stockant en tant [qu'état sérialisé](#) dans l' [entrée d'historique de session](#) appropriée . Ensuite, [StructuredDeserialize](#) est utilisé pour que la `history.state` propriété puisse renvoyer un clone de l'objet d'état fourni à l'origine.

`broadcastChannel.postMessage()` utilise [StructuredSerialize](#) sur son entrée, puis utilise [StructuredDeserialize](#) plusieurs fois sur le résultat pour produire un nouveau clone pour chaque destination diffusée. Notez que le transfert n'a pas de sens dans les situations multi-destinations.

Toute API permettant de conserver les valeurs JavaScript dans le système de fichiers utiliserait également [StructuredSerializeForStorage](#) sur son entrée et [StructuredDeserialize](#) sur sa sortie.

En général, les sites d'appel peuvent transmettre des valeurs Web IDL au lieu de valeurs JavaScript ; cela doit être compris pour effectuer une [conversion](#) implicite vers la valeur JavaScript avant d'invoquer ces algorithmes.

---

Les sites d'appel qui ne sont pas appelés à la suite d'un appel synchrone du code de l'auteur dans une méthode d'agent utilisateur doivent veiller à se [préparer](#)



[correctement à l'exécution du script](#) et [à l'exécution d'un rappel](#) avant d'appeler les opérations abstraites [StructuredSerialize](#) , [StructuredSerializeForStorage](#) ou [StructuredSerializeWithTransfer](#) , si elles sont en cours d'exécution sur des objets arbitraires. Cela est nécessaire car le processus de sérialisation peut invoquer des accesseurs définis par l'auteur dans le cadre de ses étapes finales de sérialisation profonde, et ces accesseurs pourraient appeler des opérations qui reposent sur la configuration correcte des concepts d'entrée et d'opérateur [historique](#) .

`window.postMessage()` exécute [StructuredSerializeWithTransfer](#) sur ses arguments, mais veille à le faire immédiatement, dans la partie synchrone de son algorithme. Ainsi, il est capable d'utiliser les algorithmes sans avoir besoin de [se préparer à exécuter un script](#) et [de se préparer à exécuter un rappel](#) . En revanche, une API hypothétique qui utilisait [StructuredSerialize](#) pour sérialiser périodiquement un objet fourni par l'auteur, directement à partir d'une [tâche](#) sur la [boucle d'événement](#) , devrait s'assurer qu'elle effectue au préalable les préparations appropriées. À ce jour, nous ne connaissons aucune API de ce type sur la plate-forme ; il est généralement plus simple d'effectuer la sérialisation à l'avance, en tant que conséquence synchrone du code de l'auteur.

### 2.7.10 API de clonage structuré

```
result = self.structuredClone(value[, { transfer }])
```

Prend la valeur d'entrée et renvoie une copie complète en exécutant l'algorithme de clonage structuré. [Les objets transférables](#) répertoriés dans le `transfer` tableau sont transférés, pas seulement clonés, ce qui signifie qu'ils ne sont plus utilisables dans la valeur d'entrée.

Lève un `"DataCloneError"` [DOMException](#) si une partie de la valeur d'entrée n'est pas [sérialisable](#) .



Les étapes de la méthode sont : `structuredClone(value, options)`

1. Soit *sérialisé* ? [StructuredSerializeWithTransfer](#) ( *valeur* , *options* [" `transfer` "]).
2. Soit *deserializeRecord* être ? [StructuredDeserializeWithTransfer](#) ( *sérialisé* , *c'est* le [domaine pertinent](#) ).
3. Renvoie *deserializeRecord* .[[Deserialized]].

## 3 Sémantique, structure et API des documents HTML



## 3.1 Documents

Chaque document XML et HTML dans un HTML UA est représenté par un Document objet. [\[DOM\]](#)

L' URLDocument de l'objet est définie dans *DOM* . Il est initialement défini lors de la création de l'objet, mais peut changer pendant la durée de vie de l' objet ; par exemple, il change lorsque l'utilisateur [accède](#) à un [fragment](#) de la page et lorsque la méthode est appelée avec une nouvelle URL . [\[DOM\]DocumentDocumentpushState\(\)](#)

**Les agents utilisateurs interactifs exposent généralement l' URLDocument de l'objet dans leur interface utilisateur. Il s'agit du principal mécanisme par lequel un utilisateur peut savoir si un site tente d'usurper l'identité d'un autre.**

L' origineDocument de l'objet est définie dans *DOM* . Il est initialement défini lors de la création de l'objet et peut changer au cours de la durée de vie du uniquement lors de la définition de . L'origine de A peut différer de l' origine de son URL ; par exemple, lorsqu'un [élément navigable enfant](#) est créé , l'origine de son document actif est héritée de l'origine du document actif de son parent , même si l'URL de son document actif est . [\[DOM\]DocumentDocumentdocument.domainDocumentabout:blank](#)

Lorsqu'un Document est créé par un [script](#) à l'aide des méthodes [createDocument\(\)](#) ou [createHTMLDocument\(\)](#) , le Document est immédiatement [prêt pour les tâches de post-chargement](#) .

**Le référent du document** est une chaîne (représentant une URL ) qui peut être définie lors de Document la création du. S'il n'est pas explicitement défini, sa valeur est la chaîne vide.

### 3.1.1 L' Document objet



*DOM* définit une Document interface, que cette spécification étend de manière significative.

```
enum DocumentReadyState { "loading", "interactive", "complete" };
```

```
enum DocumentVisibilityState { "visible", "hidden" };
```

```
typedef (HTMLScriptElement or SVGScriptElement)
```

```
HTMLOrSVGScriptElement;
```

```
[LegacyOverrideBuiltIns]
```

```
partial interface Document {
```

```
  // resource metadata management
```

```
  [PutForwards=href, LegacyUnforgeable] readonly attribute
```

```
    Location? location;
```

```
  attribute USVString domain;
```

```
  readonly attribute USVString referrer;
```

```
  attribute USVString cookie;
```

```
  readonly attribute DOMString lastModified;
```

```
  readonly attribute DocumentReadyState readyState;
```

```
  // DOM tree accessors
```

```
    getter object (DOMString name);
```

```
  [CEReactions] attribute DOMString title;
```

```
  [CEReactions] attribute DOMString dir;
```

```
  [CEReactions] attribute HTMLElement? body;
```

```
  readonly attribute HTMLHeadElement? head;
```

```
  [SameObject] readonly attribute HTMLCollection images;
```

```
  [SameObject] readonly attribute HTMLCollection embeds;
```

```
  [SameObject] readonly attribute HTMLCollection plugins;
```

```
  [SameObject] readonly attribute HTMLCollection links;
```

```
[SameObject] readonly attribute HTMLCollection forms;
```

```
[SameObject] readonly attribute HTMLCollection scripts;
```

```
NodeList getElementsByName(DOMString elementName);
```

```
readonly attribute HTMLScriptElement? currentScript; //
```

```
classic scripts in a document tree only
```

```
// dynamic markup insertion
```

```
[CEReactions] Document open(optional DOMString unused1,
```

```
optional DOMString unused2); // both arguments are ignored
```

```
WindowProxy? open(USVString url, DOMString name, DOMString  
features);
```

```
[CEReactions] undefined close();
```

```
[CEReactions] undefined write(DOMString... text);
```

```
[CEReactions] undefined writeln(DOMString... text);
```

```
// user interaction
```

```
readonly attribute WindowProxy? defaultView;
```

```
boolean hasFocus();
```

```
[CEReactions] attribute DOMString designMode;
```

```
[CEReactions] boolean execCommand(DOMString commandId, optional  
boolean showUI = false, optional DOMString value = "");
```

```
boolean queryCommandEnabled(DOMString commandId);
```

```
boolean queryCommandIndeterm(DOMString commandId);
```

```
boolean queryCommandState(DOMString commandId);
```

```
boolean queryCommandSupported(DOMString commandId);
```

```
DOMString queryCommandValue(DOMString commandId);
```

```
readonly attribute boolean hidden;
```

```
readonly attribute DocumentVisibilityState visibilityState;
```

```
// special event handler IDL attributes that only apply to
```

```
Document objects
```

```
[LegacyLenientThis] attribute EventHandler onreadystatechange;
```

```
attribute EventHandler onvisibilitychange;
```

```
// also has obsolete members
```

```
};
```

```
Document includes GlobalEventHandlers;
```

Chacun [Document](#) a un **conteneur de stratégie** (un [conteneur de stratégie](#) ), initialement un nouveau conteneur de stratégie, qui contient les stratégies qui s'appliquent au [Document](#).

Chacun [Document](#) a une **politique d'autorisations** , qui est une [politique d'autorisations](#) , qui est initialement vide.

Chacun [Document](#) a une **carte de module** , qui est une [carte de module](#) , initialement vide.

Chacun [Document](#) a une **politique d'ouverture d'origine croisée** , qui est une [politique d'ouverture d'origine croisée](#) , initialement une nouvelle politique d'ouverture d'origine croisée.

Chacun [Document](#) a un **is initial**[about:blank](#) , qui est un booléen, initialement faux.

Chacun [Document](#) a un **identifiant de navigation** , qui est un [identifiant de navigation](#) ou nul, initialement nul.

### 3.1.2 L' DocumentOrShadowRoot interface

DOM définit le DocumentOrShadowRoot mixin, que cette spécification étend.

```
partial interface mixin DocumentOrShadowRoot {
```

```
  readonly attribute Element? activeElement;
```

```
};
```

### 3.1.3 Gestion des métadonnées des ressources

document.referrer



Renvoie l' URL du Document à partir duquel l'utilisateur a navigué vers celui-ci, sauf s'il était bloqué ou qu'il n'y avait pas de tel document, auquel cas il renvoie la chaîne vide.

Le noreferrer type de lien peut être utilisé pour bloquer le référent.

L' **referrer** attribut doit renvoyer le référent du document .

document.cookie [ = value ]

Renvoie les cookies HTTP qui s'appliquent au Document. S'il n'y a pas de cookies ou si les cookies ne peuvent pas être appliqués à cette ressource, la chaîne vide sera renvoyée.

Peut être défini pour ajouter un nouveau cookie à l'ensemble de cookies HTTP de l'élément.

Si le contenu est mis en bac à sable dans une origine unique (par exemple dans un iframe avec l' sandbox attribut), un "SecurityError" DOMException sera lancé lors de l'obtention et de la définition.



MDN

L' **cookie** attribut représente les cookies de la ressource identifiée par l' URL du document .

Un Document objet qui tombe dans l'une des conditions suivantes est un **objet anti-cookie**Document :

- Un Document objet dont le contexte de navigation est nul.
- A Document dont le schéma d' URL n'est pas un schéma HTTP(S) .

Lors de l'obtention, si le document est un objet anti-cookieDocument , l'agent utilisateur doit renvoyer la chaîne vide. Sinon, si l' origineDocument de est une origine opaque , l'agent utilisateur doit lancer un " " . Sinon, l'agent utilisateur doit renvoyer la chaîne de cookies pour l' URL du document pour une API "non HTTP", décodée à l'aide du décodage UTF-8 sans BOM . [BISCUITS]SecurityError DOMException

Lors de la configuration, si le document est un objet anti-cookieDocument , l'agent utilisateur ne doit rien faire. Sinon, si l' origineDocument de est une origine opaque , l'agent utilisateur doit lancer un " " . Sinon, l'agent utilisateur doit agir comme il le ferait lors de la réception d'une chaîne set-cookie-string pour l' URL du document via une API "non HTTP", composée de la nouvelle valeur encodée en UTF-8 . [COOKIES] [ENCODAGE]SecurityError DOMException

*Étant donné que l' cookie attribut est accessible à travers les cadres, les restrictions de chemin sur les cookies ne sont qu'un outil pour aider à gérer quels cookies sont envoyés à quelles parties du site, et ne sont en aucun cas une fonction de sécurité. Le cookie getter et le setter de l'attribut accèdent de manière synchrone à l'état partagé. Puisqu'il n'y a pas de mécanisme de verrouillage, d'autres contextes de navigation dans un agent utilisateur multiprocessus peuvent modifier les cookies pendant l'exécution des scripts. Un site pourrait, par exemple, essayer de lire un cookie, incrémenter sa valeur, puis l'écrire à nouveau, en utilisant la nouvelle valeur du cookie comme identifiant unique pour la session ; si le site le fait deux fois dans deux fenêtres de navigateur différentes en même temps, il peut finir par utiliser le même identifiant "unique" pour les deux sessions, avec des effets potentiellement désastreux.*

---

document.lastModified



Renvoie la date de la dernière modification du document, telle que rapportée par le serveur, sous la forme " MM/DD/YYYY hh:mm:ss ", dans le fuseau horaire local de l'utilisateur.

Si la date de la dernière modification n'est pas connue, l'heure actuelle est renvoyée à la place.

L' `lastModified` attribut, lors de l'obtention, doit renvoyer la date et l'heure de la `Document` dernière modification du fichier source de , dans le fuseau horaire local de l'utilisateur, au format suivant :

1. Le composant mois de la date.
2. Un caractère SOLIDUS U+002F (/).
3. Le composant jour de la date.
4. Un caractère SOLIDUS U+002F (/).
5. Le composant année de la date.
6. Un caractère ESPACE U+0020.
7. Le composant heures du temps.
8. Un caractère U+003A COLON (:).
9. La composante minutes de l'heure.
10. Un caractère U+003A COLON (:).
11. Composante des secondes de l'heure.

Toutes les composantes numériques ci-dessus, autres que l'année, doivent être données sous la forme de deux [chiffres ASCII](#) représentant le nombre en base dix, complétés par des zéros si nécessaire. [L'année doit être donnée sous la forme de la chaîne la plus courte possible de quatre chiffres ASCII](#) ou plus représentant le nombre en base dix, complété par des zéros si nécessaire.

La `Document` date et l'heure de la dernière modification du fichier source de doivent être dérivées des caractéristiques pertinentes des protocoles de réseau utilisés, par exemple de la valeur de l'en-tête HTTP `` `Last-Modified` du document ou des métadonnées du système de fichiers pour les fichiers locaux. Si la date et l'heure de la dernière modification ne sont pas connues, l'attribut doit renvoyer la date et l'heure actuelles dans le format ci-dessus.

### 3.1.4 Signaler l'état de chargement du document

`document.readyState`

Renvoie " loading " pendant le `Document` chargement, " interactive " une fois qu'il a fini d'analyser mais charge toujours les sous-ressources, et " complete " une fois qu'il a été chargé.

L' readystatechange événement se déclenche sur l' Document objet lorsque cette valeur change.

L' DOMContentLoaded événement se déclenche après la transition vers " interactive " mais avant la transition vers " complete ", au point où toutes les sous-ressources à l'exception des async script éléments ont été chargées.



Chacun Document a une **préparation de document actuelle** , une chaîne, initialement " complete ".

Pour Document les objets créés via l' algorithme de création et d'initialisation d'un Document objet , celui-ci sera immédiatement réinitialisé à " loading " avant qu'un script puisse observer la valeur de document.readyState . Cette valeur par défaut s'applique à d'autres cas tels que s initial about:blank Document ou Documents sans contexte de navigation .

Les readyState étapes du getter consistent à retourner la préparation actuelle du document .

Pour **mettre à jour la préparation actuelle du document** pour Document document vers readinessValue :

1. Si la préparation actuelle du document du document est égale à readinessValue , alors retournez.
  2. Définit la préparation actuelle du document sur readinessValue .
  3. Si le document est associé à un analyseur HTML , alors :
    1. Soit maintenant l' heure haute résolution actuelle de l'objet global pertinent du document .
    2. Si readinessValue est " complete " et que l'heure complète DOM de l'information de synchronisation de chargement du document est 0, alors définissez l'heure complète DOM de l'information de synchronisation de chargement du document sur maintenant .
    3. Sinon, si readinessValue est " interactive " et que le temps interactif DOM des informations de synchronisation de chargement du document est 0, alors définissez le temps interactif DOM des informations de synchronisation de chargement du document sur maintenant .
  4. Déclenche un événement nommé readystatechange au document .
-



A [Document](#) est dit avoir un **parseur actif** s'il est associé à un [parseur HTML](#) ou à un [parseur XML](#) qui n'a pas encore été [arrêté](#) ou [abandonné](#) .

---

A [Document](#) a une [information de synchronisation de chargement de document](#) **info de synchronisation de chargement** .

A [Document](#) a une [information sur le délai de déchargement du document](#) **délai de déchargement du document précédent** .

A [Document](#) has a boolean **a été créé via des redirections cross-origin** , initialement false.

La [structure](#) d'informations sur le délai de chargement du document contient les [éléments](#) suivants :

***heure de début de la navigation (par défaut 0)***

Un numéro

***Temps interactif DOM (par défaut 0)***

***Heure de début de l'événement chargé par le contenu DOM (0 par défaut)***

***Heure de fin de l'événement chargé par le contenu DOM (0 par défaut)***

***Heure de fin DOM (par défaut 0)***

***charger l'heure de début de l'événement (0 par défaut)***

***charger l'heure de fin de l'événement (0 par défaut)***

[DOMHighResTimeStamp](#)valeurs

La [structure](#) d'informations sur le délai de déchargement du document contient les [éléments](#) suivants :

***décharger l'heure de début de l'événement (0 par défaut)***

***décharger l'heure de fin de l'événement (0 par défaut)***

[DOMHighResTimeStamp](#)valeurs

### ***3.1.5 Mécanisme de blocage du rendu***

Chacun [Document](#) a un **ensemble d'éléments bloquant le rendu** , un [ensemble](#) d'éléments, initialement l'ensemble vide.

Un Document *document* permet d'ajouter des éléments bloquant le rendu si le type de contenu du *document* est " " et que l'élément body du *document* est nul. text/html

Un Document *document* est **rendu bloqué** si les deux conditions suivantes sont remplies :

- l'ensemble d'éléments bloquant le rendu de *document* n'est pas vide, ou *document* autorise l'ajout d'éléments bloquant le rendu .
- L' heure de haute résolution actuelle donnée à l'objet global pertinent du *document* n'a pas dépassé une valeur de délai d'attente définie par l'implémentation .

Un élément *el* est **bloquant le rendu** si le nœud document *document* de *el* est bloqué par le rendu , et *el* est dans l'ensemble d'éléments bloquant le rendu de *document* .

Pour **bloquer le rendu** sur un élément *el* :

1. Soit *document* le nœud document de *el* .
2. Si le *document* autorise l'ajout d'éléments bloquant le rendu , alors ajoutez *el* à l'ensemble d'éléments bloquant le rendu du *document* .

Pour **débloquer le rendu** sur un élément *el* :

1. Soit *document* le nœud document de *el* .
2. Supprimez *el* de l'ensemble d'éléments bloquant le rendu du *document* .

Chaque fois qu'un élément bloquant le rendu *el* devient déconnecté du contexte de navigation ou que la valeur de l'attribut de blocage de *el* est modifiée de sorte que *el* ne soit plus potentiellement bloquant le rendu , débloquez le rendu sur *el* .

### 3.1.6 Accesseurs d'arborescence DOM

L' htmlélément d'un document est son élément document , si c'est un htmlélément, et null sinon.

---

*document* . head



Renvoie l'[head](#)élément .

L' **head**élément d'un document est le premier [head](#)élément qui est un enfant de l'[html](#)élément , s'il y en a un, ou null sinon.

L' **head**attribut, à l'obtention, doit retourner l'[head](#)élément du document (un [head](#)élément ou null).

---

```
document.title [ = value ]
```

Renvoie le titre du document, tel qu'il est donné par l'[title](#)élément pour HTML et tel qu'il est donné par l' élément [SVG](#)[title](#) pour SVG.

Peut être défini pour mettre à jour le titre du document. S'il n'y a pas d'élément approprié à mettre à jour, la nouvelle valeur est ignorée.

L' **title**élément d'un document est le premier [title](#)élément du document (dans l'[arborescence](#) ), s'il y en a un, ou nul sinon.



MDN

L' **title**attribut doit, lors de l'obtention, exécuter l'algorithme suivant :

1. Si l' [élément document](#) est un élément [SVG](#)[svg](#) , alors laissez *value* être le [contenu textuel enfant](#) du premier élément [SVG](#) [title](#) qui est un enfant de l' [élément document](#) .
2. Sinon, laissez *value* être le [contenu du texte enfant](#) de l' [title](#)élément , ou la chaîne vide si l' [title](#) élément est nul.
3. [Supprime et réduit les espaces blancs ASCII](#) dans *value* .
4. *Valeur* de retour .

Au réglage, les étapes correspondant à la première condition correspondante dans la liste suivante doivent être exécutées :

Si l' [élément du document](#) est un élément [SVG](#)[svg](#)

1. S'il existe un élément [SVG](#)[title](#) qui est un enfant de l' [élément document](#) , laissez *element* être le premier de ces éléments.
2. Sinon:

1. Soit *element* le résultat de [la création d'un élément](#) étant donné le nœud [document](#) , , et l' [espace de noms SVG de l'élément document](#) [.title](#)
2. Insérer *l'élément* comme [premier enfant](#) de l' [élément document](#) .
3. [String remplace tout](#) par la valeur donnée dans *element* .

Si l' [élément de document](#) est dans l' [espace de noms HTML](#)

1. Si l' [titleélément](#) est nul et que l' [head élément](#) est nul, alors retournez.
2. Si l' [titleélément](#) n'est pas nul, laissez *element* être l' [titleélément](#) .
3. Sinon:
  1. Soit *element* le résultat de [la création d'un élément](#) étant donné le [nœud document](#) de l'élément document , et l' [espace de noms HTML](#) [.title](#)
  2. [Ajouter](#) un élément à l' [headélément](#) .
4. [String remplace tout](#) par la valeur donnée dans *element* .

**Sinon**

Ne fais rien.

---

```
document.body [ = value ]
```

✓

Renvoie [l'élément body](#) .

Peut être défini, pour remplacer [l'élément de corps](#) .

Si la nouvelle valeur n'est pas un élément [body](#) ou [frameset](#), cela lancera un ["HierarchyRequestError"](#) [DOMException](#) .

**L'élément body** d'un document est le premier des [enfants de html](#) l'élément qui est soit un [body](#)élément, soit un [frameset](#) élément, ou null s'il n'y a pas un tel élément.

L' **body**attribut, lors de l'obtention, doit renvoyer [l'élément body](#) du document (soit un [body](#) élément, soit un [frameset](#)élément, soit null). Au réglage, l'algorithme suivant doit être exécuté :

1. Si la nouvelle valeur n'est pas un élément [body](#) ou [frameset](#), lancez un ["HierarchyRequestError"](#) [DOMException](#) .

2. Sinon, si la nouvelle valeur est la même que [celle de l'élément body](#) , return.
3. Sinon, si [l'élément body](#) n'est pas nul, [remplacez l'élément body](#) par la nouvelle valeur dans le parent de l' [élément body](#) et retournez.
4. Sinon, s'il n'y a pas [d'élément document](#) , lancer un `"HierarchyRequestError"` [DOMException](#) .
5. Sinon, [l'élément body](#) est nul, mais il existe un [élément document](#) . [Ajoutez](#) la nouvelle valeur à l' [élément de document](#) .

*La valeur renvoyée par le [body](#) getter n'est pas toujours celle transmise au setter.*

Dans cet exemple, le setter insère avec succès un [body](#) élément (bien que ce ne soit pas conforme puisque SVG n'autorise pas un [body](#) comme enfant de [SVG](#) [svg](#) ). Cependant, le getter renverra null car l'élément de document n'est pas [html](#).

```
<svg xmlns="http://www.w3.org/2000/svg">
  <script>
    document.body =
document.createElementNS("http://www.w3.org/1999/xhtml", "body");
    console.assert(document.body === null);
  </script>
</svg>
```

---

**[document.images](#)**



Renvoie un [HTMLCollection](#) des [img](#) éléments du [Document](#).

**[document.embeds](#)**



**[document.plugins](#)**



Renvoie un [HTMLCollection](#) des [embed](#) éléments du [Document](#).

**[document.links](#)**



Renvoie un [HTMLCollection](#) des éléments [a](#) et [area](#) dans le [Document](#) qui ont [href](#) des attributs.

**[document.forms](#)**



Renvoie un HTMLCollection des forméléments du Document.

`document.scripts`



Renvoie un HTMLCollection des scriptéléments du Document.

L' **images** attribut doit renvoyer un HTMLCollection enraciné au Document nœud, dont le filtre ne correspond qu'aux img éléments.

L' **embeds** attribut doit renvoyer un HTMLCollection enraciné au Document nœud, dont le filtre ne correspond qu'aux embed éléments.

L' **plugins** attribut doit renvoyer le même objet que celui renvoyé par l' embeds attribut.

L' **links** attribut doit renvoyer un HTMLCollection enraciné au Document nœud, dont le filtre correspond uniquement aux éléments avec href attributs et area aux éléments avec href attributs.

L' **forms** attribut doit renvoyer un HTMLCollection enraciné au Document nœud, dont le filtre ne correspond qu'aux form éléments.

L' **scripts** attribut doit renvoyer un HTMLCollection enraciné au Document nœud, dont le filtre ne correspond qu'aux script éléments.

---

```
collection = document.getElementsByName(name)
```



Renvoie un NodeList des éléments dans le Document qui ont un `name` attribut avec la valeur `name` .

Les étapes de la méthode consistent à retourner un live contenant tous les éléments HTML de ce document qui ont un attribut dont la valeur est identique à l' argument `elementName` , dans l'ordre de l'arborescence . Lorsque la méthode est de nouveau invoquée sur un objet avec le même argument, l'agent utilisateur peut renvoyer le même que l'objet renvoyé par l'appel précédent. Dans d'autres cas, un nouvel objet doit être retourné.  
`getElementsByName (elementName) NodeListnameDocumentNodeList`

---

## `document.currentScript`



Renvoie l' [script](#) élément, ou l' élément [SVGscript](#) , en cours d'exécution, tant que l'élément représente un [script classique](#) . Dans le cas d'une exécution de script réentrant, renvoie celui qui a commencé à s'exécuter le plus récemment parmi ceux qui n'ont pas encore fini de s'exécuter.

Renvoie null si [Document](#) n'exécute pas actuellement un élément [SVGscript](#) ou (par exemple, parce que le script en cours d'exécution est un gestionnaire d'événements ou un délai d'attente), ou si l' élément [SVG](#) ou en cours d'exécution représente un [script de module](#) . [scriptscriptscript](#)

L' `currentScript` attribut, lors de l'obtention, doit renvoyer la valeur à laquelle il a été défini le plus récemment. Lorsque le [Document](#) est créé, le `currentScript` doit être initialisé à null.

*Cette API est tombée en disgrâce dans la communauté des implémenteurs et des standards, car elle expose globalement [script](#) des éléments [SVGscript](#) . En tant que tel, il n'est pas disponible dans des contextes plus récents, comme lors de l'exécution [de scripts de module](#) ou lors de l'exécution de scripts dans une [arborescence fantôme](#) . Nous cherchons à créer une nouvelle solution pour identifier le script en cours d'exécution dans de tels contextes, ce qui ne le rend pas disponible dans le monde entier : voir [le problème #1013](#) .*

---

L' [Document](#) interface [prend en charge les propriétés nommées](#) . Les [noms de propriété pris en charge](#) d'un `document` [Document](#) d'objet à tout moment se composent des éléments suivants, dans [l'ordre de l'arborescence](#) selon l'élément qui les a contribué, en ignorant les doublons ultérieurs, et avec les valeurs des attributs venant avant les valeurs des attributs lorsque le même élément contribue aux deux : `id` `name`

- la valeur de l' `name` attribut content pour tous les éléments [exposés](#) `embed` , `form` , et [exposés](#) `iframe` qui ont un attribut content non vide et qui se trouvent [dans une arborescence de documents](#) avec `document` comme [racine](#) ; `img` `object` `name`
- la valeur de l' `id` attribut content pour tous les éléments [exposés](#) `object` qui ont un attribut content non vide `id` et qui se trouvent [dans une arborescence de documents](#) avec `document` comme [racine](#) ; et
- la valeur de l' `id` attribut content pour tous `img` les éléments qui ont à la fois un attribut content non vide `id` et un attribut content non vide `name`, et qui se

trouvent [dans une arborescence de documents](#) avec *document* comme [racine](#) .

Pour [déterminer la valeur d'un](#) *nom* de propriété nommé pour un [Document](#), l'agent utilisateur doit renvoyer la valeur obtenue en procédant comme suit :

1. Soit *elements* la liste des [éléments nommés](#) avec le nom *name* qui se trouvent [dans une arborescence de documents](#) avec the [Document](#) comme [racine](#) .

*Il y aura au moins un tel élément, par définition.*

2. Si *elements* n'a qu'un seul élément, et que cet élément est un [iframe](#) élément, et que [le contenu navigable](#)[iframe](#) de cet élément n'est pas nul, alors retournez l' [actif](#) du [contenu navigable](#) de l'élément [.WindowProxy](#)
3. Sinon, si *elements* n'a qu'un seul élément, retournez cet élément.
4. Sinon, renvoyez un [HTMLCollection](#) enraciné au [Document](#) nœud, dont le filtre ne correspond qu'aux [éléments nommés](#) avec le nom *name* .

**Les éléments nommés** avec le nom *name* , pour les besoins de l'algorithme ci-dessus, sont ceux qui sont soit :

- [Éléments exposés](#) [embed](#) , [form](#), [iframe](#), [img](#) ou [exposés](#) [object](#) qui ont un `name` attribut de contenu dont la valeur est *name* , ou
- [Les éléments exposés](#) [object](#) qui ont un `id` attribut de contenu dont la valeur est *name* , ou
- [img](#) les éléments qui ont un `id` attribut de contenu dont la valeur est *name* , et qui ont également un attribut de contenu non vide `name` présent.

Un élément [embed](#) ou [object](#) est dit exposé s'il n'a pas d'ancêtre [exposé](#) [object](#) , et, pour [object](#) les éléments, soit n'affiche pas son [contenu de secours](#) , soit n'a pas de descendants [object](#) ou [embed](#)

---

*L' [dir](#) attribut sur l' [Document](#) interface est défini avec l' [dir](#) attribut content.*

## 3.2 Éléments



### 3.2.1 Sémantique

Les éléments, les attributs et les valeurs d'attributs en HTML sont définis (par cette spécification) pour avoir certaines significations (sémantique). Par exemple, l' `ol` élément représente une liste ordonnée et l' `lang` attribut représente la langue du contenu.

Ces définitions permettent aux processeurs HTML, tels que les navigateurs Web ou les moteurs de recherche, de présenter et d'utiliser des documents et des applications dans une grande variété de contextes que l'auteur n'aurait peut-être pas pris en compte.

À titre d'exemple simple, considérons une page Web écrite par un auteur qui n'a considéré que les navigateurs Web des ordinateurs de bureau :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>My Page</title>
  </head>
  <body>
    <h1>Welcome to my page</h1>
    <p>I like cars and lorries and have a big Jeep!</p>
    <h2>Where I live</h2>
    <p>I live in a small hut on a mountain!</p>
  </body>
</html>
```

Parce que HTML transmet *du sens* plutôt que de la présentation, la même page peut également être utilisée par un petit navigateur sur un téléphone mobile, sans aucune modification de la page. Au lieu que les titres soient en grosses lettres comme sur le bureau, par exemple, le navigateur du téléphone mobile peut utiliser la même taille de texte pour toute la page, mais avec les titres en gras.

Mais cela va plus loin que les différences de taille d'écran : la même page pourrait également être utilisée par un utilisateur aveugle utilisant un navigateur basé sur la synthèse vocale, qui au lieu d'afficher la page sur un écran, lit la page à l'utilisateur, par exemple à l'aide d'un casque . Au lieu d'un texte de grande taille pour les titres, le navigateur vocal peut utiliser un volume différent ou une voix plus lente.

Ce n'est pas tout non plus. Étant donné que les navigateurs savent quelles parties de la page sont les en-têtes, ils peuvent créer un plan de document que l'utilisateur peut utiliser pour naviguer rapidement dans le document, en utilisant les touches pour "passer à l'en-tête suivant" ou "passer à l'en-tête précédent". De telles fonctionnalités sont particulièrement courantes avec les navigateurs vocaux, où les utilisateurs trouveraient autrement assez difficile de naviguer rapidement sur une page.

Même au-delà des navigateurs, les logiciels peuvent utiliser ces informations. Les moteurs de recherche peuvent utiliser les titres pour indexer plus efficacement une page ou pour fournir des liens rapides vers des sous-sections de la page à partir de leurs résultats. Les outils peuvent utiliser les en-têtes pour créer une table des matières (c'est en fait ainsi que la table des matières de cette spécification est générée).

Cet exemple s'est concentré sur les en-têtes, mais le même principe s'applique à toute la sémantique en HTML.

Les auteurs ne doivent pas utiliser d'éléments, d'attributs ou de valeurs d'attribut à des fins autres que leur objectif sémantique approprié, car cela empêche le logiciel de traiter correctement la page.

Par exemple, l'extrait de code suivant, destiné à représenter le titre d'un site d'entreprise, n'est pas conforme car la deuxième ligne n'est pas destinée à être un titre de sous-section, mais simplement un sous-titre ou un sous-titre (un titre subordonné pour la même section ).

```
<body>
  <h1>ACME Corporation</h1>
  <h2>The leaders in arbitrary fast delivery since 1920</h2>
  ...
```

L' [hgroup](#) élément peut être utilisé dans les situations suivantes :

```
<body>
  <hgroup>
    <h1>ACME Corporation</h1>
    <p>The leaders in arbitrary fast delivery since 1920</p>
  </hgroup>
  ...
```

Le document dans cet exemple suivant est également non conforme, bien qu'il soit syntaxiquement correct, car les données placées dans les cellules ne sont clairement pas des données tabulaires et l' [cite](#) élément est mal utilisé :

```
<!DOCTYPE HTML>
<html lang="en-GB">
  <head> <title> Demonstration </title> </head>
  <body>
    <table>
      <tr> <td> My favourite animal is the cat. </td> </tr>
      <tr>
        <td>
          -<a
href="https://example.org/~ernest/"><cite>Ernest</cite></a> ,
```

```

        in an essay from 1992
    </td>
</tr>
</table>
</body>
</html>

```

Cela ferait échouer le logiciel qui s'appuie sur cette sémantique : par exemple, un navigateur vocal qui permettait à un utilisateur aveugle de naviguer dans les tableaux du document signifierait la citation ci-dessus comme un tableau, ce qui confondrait l'utilisateur ; de même, un outil qui extrayait les titres des œuvres des pages extraierait "Ernest" comme titre d'une œuvre, même s'il s'agit en fait du nom d'une personne, pas d'un titre.

Une version corrigée de ce document pourrait être :

```

<!DOCTYPE HTML>
<html lang="en-GB">
  <head> <title> Demonstration </title> </head>
  <body>
    <blockquote>
      <p> My favourite animal is the cat. </p>
    </blockquote>
    <p>
      -<a href="https://example.org/~ernest/">Ernest</a>,
      in an essay from 1992
    </p>
  </body>
</html>

```

Les auteurs ne doivent pas utiliser d'éléments, d'attributs ou de valeurs d'attribut qui ne sont pas autorisés par cette spécification ou [d'autres spécifications applicables](#) , car cela complique considérablement l'extension du langage à l'avenir.

Dans l'exemple suivant, il y a une valeur d'attribut non conforme ("moquette") et un attribut non conforme ("texture"), ce qui n'est pas autorisé par cette spécification :

```

<label>Carpet: <input type="carpet" name="c" texture="deep
pile"></label>

```

Voici une manière alternative et correcte de marquer ceci:

```

<label>Carpet: <input type="text" class="carpet" name="c" data-
texture="deep pile"></label>

```

Les nœuds DOM dont [le contexte de navigation](#) du [document de nœud](#) est nul sont exemptés de toutes les exigences de conformité de document autres que les exigences [de syntaxe HTML](#) et les exigences [de syntaxe XML](#) .

En particulier, [le contexte de navigation](#) du [document de nœud du contenu](#) du modèle [template](#) de l'élément est nul. Par exemple, les exigences [du modèle de contenu](#) et les exigences de la microsyntaxe des valeurs d'attribut ne s'appliquent pas au [contenu du modèle](#) d'un élément . Dans cet exemple, un élément a des valeurs d'attribut qui sont des espaces réservés qui seraient invalides en dehors d'un élément. [templateimgtemplate](#)

```
<template>
  <article>
    
    <h1></h1>
  </article>
</template>
```

Cependant, si le balisage ci-dessus devait omettre la `</h1>` balise de fin, cela constituerait une violation de la [syntaxe HTML](#) et serait donc signalé comme une erreur par les vérificateurs de conformité.

Grâce à des scripts et à l'utilisation d'autres mécanismes, les valeurs des attributs, du texte et, en fait, de la structure entière du document peuvent changer dynamiquement pendant qu'un agent utilisateur le traite. La sémantique d'un document à un instant dans le temps est celle représentée par l'état du document à cet instant dans le temps, et la sémantique d'un document peut donc évoluer dans le temps. Les agents utilisateurs doivent mettre à jour leur présentation du document au fur et à mesure que cela se produit.

HTML a un [progress](#) élément qui décrit une barre de progression. Si son attribut "value" est mis à jour dynamiquement par un script, l'UA mettrait à jour le rendu pour montrer la progression changeante.

### 3.2.2 Éléments dans le DOM

Les nœuds représentant [des éléments HTML](#) dans le DOM doivent implémenter et exposer aux scripts les interfaces répertoriées pour eux dans les sections pertinentes de cette spécification. Cela inclut [les éléments HTML](#) dans [les documents XML](#) , même lorsque ces documents se trouvent dans un autre contexte (par exemple, à l'intérieur d'une transformation XSLT).

Les éléments du DOM **représentent** des choses ; c'est-à-dire qu'ils ont *une signification* intrinsèque , également connue sous le nom de sémantique.

Par exemple, un [ol](#) élément représente une liste ordonnée.

Les éléments peuvent être **référéncés** (référéncés) d'une manière ou d'une autre, explicitement ou implicitement. Une façon de référer explicitement un élément dans le DOM consiste à attribuer un [id](#) attribut à l'élément, puis à créer un [lien hypertexte](#) avec [id](#) la valeur de cet attribut comme [fragment](#) de la valeur d'attribut du [lien hypertexte](#) . [href](#) Les hyperliens ne sont pas nécessaires pour une référence, cependant; n'importe quelle manière de se référer à l'élément en question suffira.

Considérez l'élément suivant [figure](#), auquel est attribué un [id](#) attribut :

```
<figure id="module-script-graph">
  
  <figcaption>Figure 27: a simple module graph</figcaption>
</figure>
```

Une [référence](#) basée sur [un lien hypertexte](#) pourrait être créée à l'aide de l' élément, comme ceci :[a](#)

```
As we can see in <a href="#module-script-graph">figure 27</a>, ...
```

Cependant, il existe de nombreuses autres façons de [référer](#) l' [figure](#) élément, telles que :

- "Comme représenté sur la figure des modules A, B, C et D..."
- "Dans la Figure 27..." (sans lien hypertexte)
- "D'après le contenu de la figure 'graphique de module simple'..."
- "Dans la figure ci-dessous..." (mais [c'est déconseillé](#) )

L'interface de base, dont héritent toutes les interfaces des [éléments HTML](#) , et qui doit être utilisée par les éléments qui n'ont pas d'exigences supplémentaires, est l' [HTMLElement](#) interface.



```
[Exposed=Window]
```

```
interface HTMLElement : Element {
```

```
  [HTMLConstructor] constructor();
```

```
// metadata attributes
```

```
[CEReactions] attribute DOMString title;
```

```
[CEReactions] attribute DOMString lang;
```

```
[CEReactions] attribute boolean translate;
```

```
[CEReactions] attribute DOMString dir;
```

```
// user interaction
```

```
[CEReactions] attribute (boolean or unrestricted double or  
DOMString)? hidden;
```

```
[CEReactions] attribute boolean inert;
```

```
undefined click();
```

```
[CEReactions] attribute DOMString accessKey;
```

```
readonly attribute DOMString accessKeyLabel;
```

```
[CEReactions] attribute boolean draggable;
```

```
[CEReactions] attribute boolean spellcheck;
```

```
[CEReactions] attribute DOMString autocapitalize;
```

```
[CEReactions] attribute [LegacyNullToEmptyString] DOMString  
innerText;
```

```
[CEReactions] attribute [LegacyNullToEmptyString] DOMString  
outerText;
```

```
ElementInternals attachInternals();
```

```
// The popover API
```

```
undefined showPopover\(\) ;
```

```
undefined hidePopover\(\) ;
```

```
undefined togglePopover(optional boolean force);
```

```
[CEReactions] attribute DOMString? popover;
```

```
};
```

```
HTMLElement includes GlobalEventHandlers;
```

```
HTMLElement includes ElementContentEditable;
```

```
HTMLElement includes HTMLOrSVGElement;
```

```
[Exposed=Window]
```

```
interface HTMLUnknownElement : HTMLElement {
```

```
// Note: intentionally no [HTMLConstructor]
```

```
};
```

L' [HTMLElement](#) interface contient des méthodes et des attributs liés à un certain nombre de fonctionnalités disparates, et les membres de cette interface sont donc décrits dans différentes sections de cette spécification.

---

L' [interface d'élément](#) pour un élément portant le nom *name* dans l' [espace de noms HTML](#) est déterminée comme suit :

1. Si le *nom* est [applet](#), [bgsound](#), [blink](#), [isindex](#), [keygen](#), [multicol](#), [nextid](#), ou [spacer](#), alors retourne [HTMLUnknownElement](#).
2. Si le *nom* est [acronym](#), [basefont](#), [big](#), [center](#), [nobr](#), [noembed](#), [noframes](#), [plaintext](#), [rb](#), [rtc](#), [strike](#), ou [tt](#), alors retourne [HTMLElement](#).

3. Si le nom est [listing](#) ou [xmp](#), alors retourne [HTMLPreElement](#).
4. Sinon, si cette spécification définit une interface appropriée pour le [type d'élément](#) correspondant au nom local *name*, alors retournez cette interface.
5. Si [d'autres spécifications applicables](#) définissent une interface appropriée pour *name*, retournez l'interface qu'elles définissent.
6. Si *name* est un [nom d'élément personnalisé valide](#), alors return [HTMLElement](#).
7. Retour [HTMLUnknownElement](#).

*L'utilisation de [HTMLElement](#) au lieu de [HTMLUnknownElement](#) dans le cas de [noms d'éléments personnalisés valides](#) est faite pour s'assurer que toute mise à [niveau](#) future potentielle ne provoque qu'une transition linéaire de la chaîne de prototypes de l'élément, d' [HTMLElement](#) une sous-classe, au lieu d'une chaîne latérale, d' [HTMLUnknownElement](#) une sous-classe non liée.*

Les fonctionnalités partagées entre les éléments HTML et SVG utilisent l' [HTMLOrSVGElement](#) interface mixin : [\[SVG\]](#)

```
interface mixin HTMLOrSVGElement {
```

```
  [SameObject] readonly attribute DOMStringMap dataset;
```

```
  attribute DOMString nonce; // intentionally no \[CEReactions\]
```

```
  [CEReactions] attribute boolean autofocus;
```

```
  [CEReactions] attribute long tabIndex;
```

```
  undefined focus(optional FocusOptions options = {});
```

```
  undefined blur();
```

```
};
```

Un exemple d'élément qui n'est ni un élément HTML ni un élément SVG est créé comme suit :

```
const el = document.createElementNS("some namespace", "example");
console.assert(el.constructor === Element);
```



### 3.2.3 Constructeurs d'éléments HTML

Pour prendre en charge la fonctionnalité [des éléments personnalisés](#), tous les éléments HTML ont un comportement de constructeur spécial. Ceci est indiqué via l' [attribut étendu](#) `[HTMLConstructor]` IDL. Il indique que l'objet d'interface pour l'interface donnée aura un comportement spécifique lorsqu'il sera appelé, comme défini en détail ci-dessous.

L' `[HTMLConstructor]` attribut étendu ne doit prendre aucun argument et ne doit apparaître que sur [les opérations du constructeur](#). Il ne doit apparaître qu'une seule fois sur une opération de constructeur, et l'interface ne doit contenir qu'une seule opération de constructeur annotée, et aucune autre. L'opération de constructeur annotée doit être déclarée pour ne prendre aucun argument.

Les interfaces déclarées avec des opérations de constructeur qui sont annotées avec l' attribut extended ont les [étapes de constructeur](#) [remplacées](#) `[HTMLConstructor]` suivantes :

1. Soit *le registre* l' objet de [l'objet global actuel](#) `CustomElementRegistry`.
2. Si [NewTarget](#) est égal à l' [objet de fonction actif](#), lancez un `TypeError`.

Cela peut se produire lorsqu'un élément personnalisé est défini à l'aide d'une [interface d'élément](#) comme constructeur :

```
customElements.define("bad-1", HTMLButtonElement);  
new HTMLButtonElement(); // (1)  
document.createElement("bad-1"); // (2)
```

Dans ce cas, lors de l'exécution de `HTMLButtonElement` (soit explicitement, comme dans (1), soit implicitement, comme dans (2)), l' [objet fonction actif](#) et [NewTarget](#) sont `HTMLButtonElement`. Si cette vérification n'était pas présente, il serait possible de créer une instance dont `HTMLButtonElement` le nom local serait `bad-1`.

3. Soit *definition* l'entrée dans *le registre* avec [un constructeur](#) égal à [NewTarget](#). S'il n'y a pas une telle définition, lancez un `TypeError`.

*Puisqu'il ne peut y avoir aucune entrée dans le registre avec un [constructeur](#) indéfini, cette étape empêche également les constructeurs d'éléments HTML d'être appelés en tant que fonctions (puisque dans ce cas [NewTarget](#) sera indéfini).*

4. Soit sa valeur nulle.
5. Si [le nom local](#) de la définition est égal au [nom](#) de la définition (c'est-à-dire que la définition concerne un [élément personnalisé autonome](#)), alors :

1. Si l' [objet de fonction actif](#) n'est pas [HTMLElement](#), lancez un [TypeError](#).

Cela peut se produire lorsqu'un élément personnalisé est défini pour ne pas étendre les noms locaux, mais hérite d'une non- [HTMLElement](#) classe :

```
customElements.define("bad-2", class Bad2 extends HTMLParagraphElement {});
```

Dans ce cas, lors de l'appel (implicite) `super()` qui se produit lors de la construction d'une instance de `Bad2`, l' [objet fonction actif](#) est [HTMLParagraphElement](#), et non [HTMLElement](#).

6. Sinon (c'est-à-dire si *la définition* concerne un [élément intégré personnalisé](#) ) :

1. Soit *les noms locaux valides* la liste des noms locaux des éléments définis dans cette spécification ou dans [d'autres spécifications applicables](#) qui utilisent l' [objet fonction actif](#) comme [interface d'élément](#) .
2. Si *les noms locaux valides* ne contiennent pas [le nom local](#) de *la définition* , lancez un [TypeError](#)

Cela peut se produire lorsqu'un élément personnalisé est défini pour étendre un nom local donné mais hérite de la mauvaise classe :

```
customElements.define("bad-3", class Bad3 extends HTMLQuoteElement {}, { extends: "p" });
```

Dans ce cas, lors de l' `super()` appel (implicite) qui se produit lors de la construction d'une instance de `Bad3`, *les noms locaux valides* sont la liste contenant `get` [blockquote](#), mais [le nom local](#) de *la définition* est , qui n'est pas dans cette liste.`p`

3. Set est la valeur du [nom](#) de *la définition* .

7. Si [la pile de construction](#) de *la définition* est vide, alors :

1. Soit *element* le résultat de [la création interne d'un nouvel objet implémentant l'interface](#) à laquelle correspond l' [objet de fonction actif](#) , étant donné le domaine actuel et [NewTarget](#) .
2. Définissez [le document de nœud](#) de l'élément sur celui [associé](#) à l'objet [global actuel](#) .[Document](#)
3. Définissez [l'espace de noms](#) de l'élément sur l' [espace de noms HTML](#) .
4. Définissez [le préfixe d'espace de noms](#) de l'élément sur null.

5. Définissez [le nom local](#) de l'élément sur [le nom local](#) de la définition .
6. Définissez [l'état de l'élément personnalisé](#) de l' élément sur " ".custom
7. Définissez la définition de l' [élément personnalisé](#) de l'élément sur *définition* .
8. Définissez [la valeur](#) de l'élément sur *is value* [.is](#)
9. *Élément* de retour .

*Cela se produit lorsque le script de l'auteur construit directement un nouvel élément personnalisé, par exemple via `new MyCustomElement()` .*

8. Que le prototype soit ? [Get](#) ( [NouvelleCible](#) , "prototype").
9. Si [Type](#) ( *prototype* ) n'est pas Objet, alors :
  1. Que le royaume soit ? [GetFunctionRealm](#) ( [NouvelleCible](#) ).
  2. Définissez *prototype* sur l' [objet prototype d'interface](#) du *domaine* dont l'interface est la même que l'interface de l' [objet de fonction actif](#) .

*Le domaine de l' [objet de la fonction active](#) peut ne pas être realm , nous utilisons donc le concept plus général de "la même interface" à travers les domaines ; nous ne recherchons pas l'égalité des [objets d'interface](#) . Ce comportement de secours, y compris l'utilisation du domaine de [NewTarget](#) et la recherche du prototype approprié, est conçu pour correspondre à un comportement analogue pour les éléments intégrés JavaScript et Web IDL [crée en interne un nouvel objet implémentant l' algorithme d'interface](#).*

10. Soit *element* la dernière entrée dans [la pile de construction](#) de la *définition* .
11. Si l' élément est un [marqueur déjà construit](#) , lancez alors un ["InvalidStateError"](#) [DOMException](#) .

Cela peut se produire lorsque le code de l'auteur à l'intérieur du [constructeur d'élément personnalisé](#) crée [de manière non conforme](#)`super()` une autre instance de la classe en cours de construction, avant d'appeler :

```
let doSillyThing = true;

class DontDoThis extends HTMLElement {
  constructor() {
    if (doSillyThing) {
      doSillyThing = false;
      new DontDoThis();

      // Now the construction stack will contain an already
      constructed marker.
    }
  }
}
```

```
// This will then fail with an "InvalidStateError"
DOMException:
    super();
  }
}
```

Cela peut également se produire lorsque le code de l'auteur à l'intérieur du [constructeur d'élément personnalisé](#) appelle deux fois de manière [non conforme](#), car selon la spécification JavaScript, cela exécute en fait le [constructeur de la superclasse \(c'est-à-dire cet algorithme\) deux fois, avant de générer une erreur](#) : `super()`

```
class DontDoThisEither extends HTMLElement {
  constructor() {
    super();

    // This will throw, but not until it has already called
    // into the HTMLElement constructor
    super();
  }
}
```

12. Effectuer ? `élément.[[SetPrototypeOf]]( prototype )`.

13. Remplacez la dernière entrée dans [la pile de construction](#) de la *définition* par un [marqueur déjà construit](#).

14. *Élément* de retour .

*Cette étape est normalement atteinte lors [de la mise à niveau](#) d'un élément personnalisé ; l'élément existant est renvoyé, de sorte que l' `super()` appel à l'intérieur du [constructeur d'élément personnalisé](#) affecte cet élément existant à **this** .*

En plus du comportement de constructeur impliqué par [\[HTMLConstructor\]](#), certains éléments ont également [des constructeurs nommés](#) (qui sont en réalité des fonctions d'usine avec une `prototype` propriété modifiée).

Les constructeurs nommés pour les éléments HTML peuvent également être utilisés dans une `extends` clause lors de la définition d'un [constructeur d'élément personnalisé](#) :

```
class AutoEmbiggenedImage extends Image {
```

```

    constructor(width, height) {
      super(width * 10, height * 10);
    }
  }

  customElements.define("auto-embiggened", AutoEmbiggenedImage, {
    extends: "img" });

  const image = new AutoEmbiggenedImage(15, 20);
  console.assert(image.width === 150);
  console.assert(image.height === 200);

```

### 3.2.4 Définitions des éléments

Chaque élément de cette spécification a une définition qui inclut les informations suivantes :

#### **Catégories**

Liste des [catégories](#) auxquelles appartient l'élément. Ceux-ci sont utilisés lors de la définition des [modèles de contenu](#) pour chaque élément.

#### **Contextes dans lesquels cet élément peut être utilisé**

Une description *non normative* de l'endroit où l'élément peut être utilisé. Ces informations sont redondantes avec les modèles de contenu des éléments qui autorisent celui-ci en tant qu'enfant, et ne sont fournies qu'à titre de commodité.

*Pour plus de simplicité, seules les attentes les plus spécifiques sont listées.*

*Par exemple, tout [le contenu de phrasé](#) est [un contenu de flux](#) . Ainsi, les éléments qui expriment [du contenu](#) ne seront répertoriés que comme "où [le contenu de phrasé](#) est attendu", car il s'agit de l'attente la plus spécifique. Partout qui attend [du contenu fluide](#) , il attend également [du contenu de phrasé](#) et répond donc également à cette attente.*

#### **Modèle de contenu**

Une description normative du contenu qui doit être inclus en tant qu'enfants et descendants de l'élément.

#### **Omission de balise dans `text/html`**

Une description *non normative* [text/html](#) indiquant si, dans la syntaxe, les balises [de début](#) et [de fin](#) peuvent être omises. Ces informations sont redondantes avec les exigences normatives données dans la section [des](#)

[balises facultatives](#) et sont fournies dans les définitions d'éléments uniquement à titre de commodité.

### **Attributs de contenu**

Une liste normative d'attributs qui peuvent être spécifiés sur l'élément (sauf indication contraire), ainsi que des descriptions non normatives de ces attributs. (Le contenu à gauche du tiret est normatif, le contenu à droite du tiret ne l'est pas.)

### **Considérations d'accessibilité**

Pour les auteurs : les exigences de conformité pour l'utilisation d' *ARIA* [role](#) et [aria-\\*](#) des attributs sont définies dans *ARIA en HTML* . [\[ARIA\]](#) [\[ARIAHTML\]](#)

Pour les implémenteurs : les exigences de l'agent utilisateur pour l'implémentation de la sémantique de l'API d'accessibilité sont définies dans *HTML Accessibility API Mappings* . [\[HTMLAAM\]](#)

### **Interface DOM**

Une définition normative d'une interface DOM que de tels éléments doivent implémenter.

Ceci est ensuite suivi d'une description de ce que l'élément [représente](#) , ainsi que de tout critère de conformité normative supplémentaire qui peut s'appliquer aux auteurs et aux implémentations. Des exemples sont parfois également inclus.

#### **3.2.4.1 Attributs**

Une valeur d'attribut est une chaîne. Sauf indication contraire, les valeurs d'attribut sur [les éléments HTML](#) peuvent être n'importe quelle valeur de chaîne, y compris la chaîne vide, et il n'y a aucune restriction sur le texte qui peut être spécifié dans ces valeurs d'attribut.

#### **3.2.5 Modèles de contenu**

Chaque élément défini dans cette spécification a un modèle de contenu : une description du [contenu](#) attendu de l'élément . Un [élément HTML](#) doit avoir un contenu qui correspond aux exigences décrites dans le modèle de contenu de l'élément. Le **contenu** d'un élément est ses enfants dans le DOM.

[Les espaces blancs ASCII](#) sont toujours autorisés entre les éléments. Les agents utilisateurs représentent ces caractères entre les éléments du balisage source en tant que [Text](#) nœuds dans le DOM. Les nœuds vides [Text](#) et [Text](#) les nœuds

constitués uniquement de séquences de ces caractères sont considérés comme **des espaces blancs inter-éléments** .

[Les espaces blancs inter-éléments](#) , les nœuds de commentaires et les nœuds d'instructions de traitement doivent être ignorés lors de l'établissement si le contenu d'un élément correspond ou non au modèle de contenu de l'élément, et doivent être ignorés lors de l'application d'algorithmes qui définissent la sémantique du document et de l'élément.

*Ainsi, un élément A est dit précédé ou suivi d'un deuxième élément B si A et B ont le même nœud parent et qu'il n'y a pas d'autres nœuds ou [Text](#)nœuds d'élément (autres que [les espaces blancs inter-éléments](#) ) entre eux. De même, un nœud est le seul enfant d'un élément si cet élément ne contient pas d'autres nœuds autres que [les espaces blancs inter-éléments](#) , les nœuds de commentaires et les nœuds d'instructions de traitement.*

Les auteurs ne doivent utiliser [des éléments HTML](#) nulle part, sauf là où ils sont explicitement autorisés, comme défini pour chaque élément, ou comme explicitement requis par d'autres spécifications. Pour les documents composés XML, ces contextes peuvent se trouver à l'intérieur d'éléments provenant d'autres espaces de noms, si ces éléments sont définis comme fournissant les contextes pertinents.

Le format de syndication Atom définit un `content`élément. Lorsque son `type`attribut a la valeur `xhtml`, le format de syndication Atom exige qu'il contienne un seul `div`élément HTML. Ainsi, un `div`élément est autorisé dans ce contexte, même si cela n'est pas explicitement énoncé de manière normative par cette spécification. [\[ATOME\]](#)

De plus, [les éléments HTML](#) peuvent être des nœuds orphelins (c'est-à-dire sans nœud parent).

Par exemple, créer un `td`élément et le stocker dans une variable globale dans un script est conforme, même si `td`les éléments ne sont autrement censés être utilisés qu'à l'intérieur `tr`des éléments.

```
var data = {  
  name: "Banana",  
  cell: document.createElement('td'),  
};
```

### 3.2.5.1 Le modèle de contenu "rien"

Lorsque le modèle de contenu d'un élément est **null** , l'élément ne doit contenir aucun [Text](#)nœud (autre que [l'espace blanc inter-élément](#) ) et aucun nœud d'élément.

*La plupart des éléments HTML dont le modèle de contenu est « rien » sont également, par commodité, des éléments vides (éléments qui n'ont pas de balise de fin dans la syntaxe HTML ). Cependant, ce sont des concepts entièrement distincts.*

### 3.2.5.2 Types de contenu

Chaque élément en HTML appartient à zéro ou plusieurs **catégories** qui regroupent des éléments ayant des caractéristiques similaires. Les grandes catégories suivantes sont utilisées dans cette spécification :

- [Contenu des métadonnées](#)
- [Contenu du flux](#)
- [Sectionnement du contenu](#)
- [Contenu de l'en-tête](#)
- [Contenu de la formulation](#)
- [Contenu intégré](#)
- [Contenu interactif](#)

*Certains éléments entrent également dans d'autres catégories, qui sont définies dans d'autres parties de cette spécification.*

Ces catégories sont liées comme suit :

Le contenu de sectionnement, le contenu d'en-tête, le contenu de formulation, le contenu intégré et le contenu interactif sont tous des types de contenu de flux. Les métadonnées sont parfois du contenu de flux. Les métadonnées et le contenu interactif expriment parfois le contenu. Le contenu intégré est également un type de contenu de phrasé et est parfois un contenu interactif.

D'autres catégories sont également utilisées à des fins spécifiques, par exemple, les contrôles de formulaire sont spécifiés à l'aide d'un certain nombre de catégories pour définir des exigences communes. Certains éléments ont des exigences uniques et ne rentrent dans aucune catégorie particulière.

#### 3.2.5.2.1 Contenu des métadonnées

**Le contenu des métadonnées** est un contenu qui établit la présentation ou le comportement du reste du contenu, ou qui établit la relation du document avec d'autres documents, ou qui transmet d'autres informations "hors bande".

- [base](#)
- [link](#)
- [meta](#)
- [noscript](#)
- [script](#)



- [style](#)
- [template](#)
- [title](#)

Les éléments d'autres espaces de noms dont la sémantique est principalement liée aux métadonnées (par exemple RDF) sont également [du contenu de métadonnées](#) .

Ainsi, dans la sérialisation XML, on peut utiliser RDF, comme ceci :

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:r="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
      xml:lang="en">
  <head>
    <title>Hedral's Home Page</title>
    <r:RDF>
      <Person xmlns="http://www.w3.org/2000/10/swap/pim/contact#"
              r:about="https://hedral.example.com/#">
        <fullName>Cat Hedral</fullName>
        <mailbox r:resource="mailto:hedral@damowmow.com"/>
        <personalTitle>Sir</personalTitle>
      </Person>
    </r:RDF>
  </head>
  <body>
    <h1>My home page</h1>
    <p>I like playing with string, I guess. Sister says squirrels are
fun
    too so sometimes I follow her to play with them.</p>
  </body>
</html>
```

Ce n'est pas possible dans la sérialisation HTML, cependant.

### 3.2.5.2.2 Contenu du flux

La plupart des éléments utilisés dans le corps des documents et des applications sont classés comme **contenu de flux** .

- [a](#)
- [abbr](#)
- [address](#)
- [area](#)(si c'est un descendant d'un [map](#)élément)
- [article](#)

- [aside](#)
- [audio](#)
- [b](#)
- [bdi](#)
- [bdo](#)
- [blockquote](#)
- [br](#)
- [button](#)
- [canvas](#)
- [cite](#)
- [code](#)
- [data](#)
- [datalist](#)
- [del](#)
- [details](#)
- [dfn](#)
- [dialog](#)
- [div](#)
- [dl](#)
- [em](#)
- [embed](#)
- [fieldset](#)
- [figure](#)
- [footer](#)
- [form](#)
- [h1](#)
- [h2](#)
- [h3](#)
- [h4](#)
- [h5](#)
- [h6](#)
- [header](#)
- [hgroup](#)
- [hr](#)
- [i](#)
- [iframe](#)
- [img](#)
- [input](#)
- [ins](#)
- [kbd](#)
- [label](#)
- [link](#)(si c'est [autorisé dans le corps](#) )
- [main](#)(si c'est un [élément hiérarchiquement correct](#)[main](#) )
- [map](#)
- [mark](#)
- [MathML](#)[math](#)
- [menu](#)
- [meta](#)(si l' [itemprop](#) attribut est présent)
- [meter](#)
- [nav](#)

- [noscript](#)
- [object](#)
- [ol](#)
- [output](#)
- [p](#)
- [picture](#)
- [pre](#)
- [progress](#)
- [q](#)
- [ruby](#)
- [s](#)
- [samp](#)
- [script](#)
- [section](#)
- [select](#)
- [slot](#)
- [small](#)
- [span](#)
- [strong](#)
- [sub](#)
- [sup](#)
- [SVG<sub>svg</sub>](#)
- [table](#)
- [template](#)
- [textarea](#)
- [time](#)
- [u](#)
- [ul](#)
- [var](#)
- [video](#)
- [wbr](#)
- [éléments personnalisés autonomes](#)
- [texte](#)

#### **3.2.5.2.3 Sectionner le contenu**

**Le contenu de sectionnement** est un contenu qui définit la portée des éléments [header](#) et [footer](#).

- [article](#)
- [aside](#)
- [nav](#)
- [section](#)

#### 3.2.5.2.4 Contenu du titre

**Le contenu de l'en-tête** définit l'en-tête d'une section (qu'il soit explicitement balisé à l'aide d'éléments [de contenu de sectionnement](#) ou implicite par le contenu de l'en-tête lui-même).

- [h1](#)
- [h2](#)
- [h3](#)
- [h4](#)
- [h5](#)
- [h6](#)
- [hgroup](#) (s'il a un descendant [h1](#) de [h6](#) l'élément)

#### 3.2.5.2.5 Phrase contenu

**Le contenu de la phrase** est le texte du document, ainsi que les éléments qui marquent ce texte au niveau intra-paragraphe. Séries de [paragraphe](#)s de forme [de contenu de phrase](#) .

- [a](#)
- [abbr](#)
- [area](#) (si c'est un descendant d'un [map](#) élément)
- [audio](#)
- [b](#)
- [bdi](#)
- [bdo](#)
- [br](#)
- [button](#)
- [canvas](#)
- [cite](#)
- [code](#)
- [data](#)
- [datalist](#)
- [del](#)
- [dfn](#)
- [em](#)
- [embed](#)
- [i](#)
- [iframe](#)
- [img](#)
- [input](#)
- [ins](#)
- [kbd](#)
- [label](#)
- [link](#) (si c'est [autorisé dans le corps](#) )
- [map](#)

- [mark](#)
- [MathML<sub>math</sub>](#)
- [meta](#)(si l' [itemprop](#) attribut est présent)
- [meter](#)
- [noscript](#)
- [object](#)
- [output](#)
- [picture](#)
- [progress](#)
- [q](#)
- [ruby](#)
- [s](#)
- [samp](#)
- [script](#)
- [select](#)
- [slot](#)
- [small](#)
- [span](#)
- [strong](#)
- [sub](#)
- [sup](#)
- [SVG<sub>svg</sub>](#)
- [template](#)
- [textarea](#)
- [time](#)
- [u](#)
- [var](#)
- [video](#)
- [wbr](#)
- [éléments personnalisés autonomes](#)
- [texte](#)

*La plupart des éléments catégorisés comme contenu de phrasé ne peuvent contenir que des éléments eux-mêmes catégorisés comme contenu de phrasé, pas n'importe quel contenu de flux.*

**Text** , dans le contexte des modèles de contenu, signifie soit rien, soit [Text](#) des nœuds. [Le texte](#) est parfois utilisé comme modèle de contenu en soi, mais il [exprime également le contenu](#) et peut être [un espace blanc inter-éléments](#) (si les [Text](#) nœuds sont vides ou contiennent uniquement [des espaces blancs ASCII](#) ).

[Text](#) les nœuds et les valeurs d'attribut doivent être constitués de [valeurs scalaires](#) , à l'exclusion [des non-caractères](#) , et [des contrôles](#) autres que [les espaces blancs ASCII](#) . Cette spécification inclut des contraintes supplémentaires sur la valeur exacte des [Text](#) nœuds et des valeurs d'attribut en fonction de leur contexte précis.

### 3.2.5.2.6 Contenu intégré

**Le contenu intégré** est un contenu qui importe une autre ressource dans le document, ou un contenu d'un autre vocabulaire qui est inséré dans le document.

- [audio](#)
- [canvas](#)
- [embed](#)
- [iframe](#)
- [img](#)
- [MathML](#)<sub>math</sub>
- [object](#)
- [picture](#)
- [SVG](#)<sub>svg</sub>
- [video](#)

Les éléments qui proviennent d'espaces de noms autres que l' [espace de noms HTML](#) et qui véhiculent du contenu mais pas des métadonnées, sont [du contenu intégré](#) pour les besoins des modèles de contenu définis dans la présente spécification. (Par exemple, MathML ou SVG.)

Certains éléments de contenu intégrés peuvent avoir **un contenu de secours** : contenu à utiliser lorsque la ressource externe ne peut pas être utilisée (par exemple parce qu'elle est d'un format non pris en charge). Les définitions d'élément indiquent ce qu'est le repli, le cas échéant.

#### **3.2.5.2.7 Contenu interactif**

**Le contenu interactif** est un contenu spécifiquement destiné à l'interaction de l'utilisateur.

- [a](#)(si l' [href](#)attribut est présent)
- [audio](#)(si l' [controls](#)attribut est présent)
- [button](#)
- [details](#)
- [embed](#)
- [iframe](#)
- [img](#)(si l' [usemap](#)attribut est présent)
- [input](#)(si l' [type](#)attribut n'est pas dans l' état [Masqué](#) )
- [label](#)
- [select](#)
- [textarea](#)
- [video](#)(si l' [controls](#)attribut est présent)

#### **3.2.5.2.8 Contenu palpable**

En règle générale, les éléments dont le modèle de contenu autorise n'importe quel [contenu de flux](#) ou [contenu de phrasé](#) doivent avoir au moins un nœud dans son [contenu](#) qui est un **contenu palpable** et qui n'a pas l' [hidden](#) attribut spécifié.

*[Le contenu palpable rend un élément non vide en fournissant soit du texte non vide descendant , soit quelque chose que les utilisateurs peuvent entendre \( \[audio\]\(#\)éléments\) ou afficher \( \[video\]\(#\)éléments \[img\]\(#\), ou \[canvas\]\(#\)éléments\) ou avec lequel interagir \(par exemple, des contrôles de formulaire interactifs\).](#)*

Cette exigence n'est cependant pas une exigence stricte, car il existe de nombreux cas où un élément peut être vide légitimement, par exemple lorsqu'il est utilisé comme espace réservé qui sera rempli ultérieurement par un script, ou lorsque l'élément fait partie d'un modèle et serait rempli sur la plupart des pages, mais sur certaines pages n'est pas pertinent.

Les vérificateurs de conformité sont encouragés à fournir un mécanisme permettant aux auteurs de trouver les éléments qui ne satisfont pas à cette exigence, en tant qu'aide à la rédaction.

Les éléments suivants constituent un contenu palpable :

- [a](#)
- [abbr](#)
- [address](#)
- [article](#)
- [aside](#)
- [audio](#)(si l' [controls](#)attribut est présent)
- [b](#)
- [bdi](#)
- [bdo](#)
- [blockquote](#)
- [button](#)
- [canvas](#)
- [cite](#)
- [code](#)
- [data](#)
- [del](#)
- [details](#)
- [dfn](#)
- [div](#)
- [dl](#)(si les enfants de l'élément incluent au moins un groupe nom-valeur)
- [em](#)
- [embed](#)
- [fieldset](#)
- [figure](#)
- [footer](#)
- [form](#)
- [h1](#)
- [h2](#)
- [h3](#)

- [h4](#)
- [h5](#)
- [h6](#)
- [header](#)
- [hgroup](#)
- [i](#)
- [iframe](#)
- [img](#)
- [input](#)(si l' [type](#)attribut n'est pas dans l' état [Masqué](#) )
- [ins](#)
- [kbd](#)
- [label](#)
- [main](#)
- [map](#)
- [mark](#)
- [MathML](#)[math](#)
- [menu](#)(si les enfants de l'élément incluent au moins un [li](#)élément)
- [meter](#)
- [nav](#)
- [object](#)
- [ol](#)(si les enfants de l'élément incluent au moins un [li](#)élément)
- [output](#)
- [p](#)
- [picture](#)
- [pre](#)
- [progress](#)
- [q](#)
- [ruby](#)
- [s](#)
- [samp](#)
- [section](#)
- [select](#)
- [small](#)
- [span](#)
- [strong](#)
- [sub](#)
- [sup](#)
- [SVG](#)[svg](#)
- [table](#)
- [textarea](#)
- [time](#)
- [u](#)
- [ul](#)(si les enfants de l'élément incluent au moins un [li](#)élément)
- [var](#)
- [video](#)
- [éléments personnalisés autonomes](#)
- [texte](#) qui n'est pas [un espace entre les éléments](#)



### 3.2.5.2.9 Éléments de support de script

**Éléments de support de script** sont ceux qui ne [représentent](#) rien eux-mêmes (c'est-à-dire qu'ils ne sont pas rendus), mais sont utilisés pour supporter les scripts, par exemple pour fournir des fonctionnalités à l'utilisateur.

Les éléments suivants sont des éléments prenant en charge les scripts :

- [script](#)
- [template](#)

### 3.2.5.3 Modèles de contenu transparents

Certains éléments sont qualifiés de **transparent** ; ils ont "transparent" dans la description de leur modèle de contenu. Le modèle de contenu d'un [transparent](#) est dérivé du modèle de contenu de son élément parent : les éléments requis dans la partie du modèle de contenu qui est "transparente" sont les mêmes éléments que ceux requis dans la partie du modèle de contenu du parent de l'élément transparent dans lequel se trouve l'élément transparent.

Par exemple, un [ins](#)élément à l'intérieur d'un [ruby](#)élément ne peut pas contenir d' [rt](#)élément, car la partie du [ruby](#)modèle de contenu de l'élément qui autorise [ins](#)les éléments est la partie qui autorise [la formulation de contenu](#) , et l' [rt](#)élément n'est pas [une formulation de contenu](#) .

*Dans certains cas, où des éléments transparents sont imbriqués les uns dans les autres, le processus doit être appliqué de manière itérative.*

Considérez le fragment de balisage suivant :

```
<p><object><param><ins><map><a  
href="/">Apples</a></map></ins></object></p>
```

Pour vérifier si "Apples" est autorisé à l'intérieur de l' [a](#)élément, les modèles de contenu sont examinés. Le [a](#)modèle de contenu de l'élément est transparent, de même que celui de l' [map](#) élément, de même que celui de l' [ins](#)élément, de même que la partie de l' [object](#) élément dans laquelle l' [ins](#)élément se trouve. L' [object](#)élément se trouve dans l' [p](#)élément dont le modèle de contenu est [phrasing content](#) . Ainsi, "Pommes" est autorisé, car le texte exprime le contenu.

Lorsqu'un élément transparent n'a pas de parent, la partie de son modèle de contenu qui est "transparente" doit à la place être traitée comme acceptant tout [contenu de flux](#) .

### 3.2.5.4 Paragraphes

*Le terme paragraphe tel que défini dans cette section est utilisé pour plus que la simple définition de l'pélément. Le concept de paragraphe défini ici est utilisé pour décrire comment interpréter les documents. L'pélément n'est qu'une des nombreuses manières de marquer un paragraphe .*

Un **paragraphe** est généralement une séquence de contenu de phrasé qui forme un bloc de texte avec une ou plusieurs phrases qui traitent d'un sujet particulier, comme dans la typographie, mais peut également être utilisé pour un regroupement thématique plus général. Par exemple, une adresse est aussi un paragraphe, tout comme une partie d'un formulaire, une signature ou une strophe dans un poème.

Dans l'exemple suivant, il y a deux paragraphes dans une section. Il y a aussi un titre, qui contient un contenu de phrase qui n'est pas un paragraphe. Notez que les commentaires et les espaces entre les éléments ne forment pas de paragraphes.

```
<section>
  <h2>Example of paragraphs</h2>
  This is the <em>first</em> paragraph in this example.
  <p>This is the second.</p>
  <!-- This is not a paragraph. -->
</section>
```

Les paragraphes dans le contenu du flux sont définis par rapport à l'apparence du document sans que les éléments a, ins, del et map ne compliquent les choses, car ces éléments, avec leurs modèles de contenu hybrides, peuvent chevaucher les limites des paragraphes, comme illustré dans les deux premiers exemples ci-dessous.

*En règle générale, il est préférable d'éviter d'avoir des éléments à cheval sur les limites des paragraphes. Maintenir un tel balisage peut être difficile.*

L'exemple suivant prend le balisage de l'exemple précédent et place insdes deléléments autour d'une partie du balisage pour montrer que le texte a été modifié (bien que dans ce cas, les modifications n'aient certes pas beaucoup de sens). Remarquez comment cet exemple a exactement les mêmes paragraphes que le précédent, malgré les éléments inset del - l' insélément chevauche le titre et le premier paragraphe, et l' delélément chevauche la limite entre les deux paragraphes.

```
<section>
  <ins><h2>Example of paragraphs</h2>
  This is the <em>first</em> paragraph in</ins> this example<del>.
  <p>This is the second.</p></del>
  <!-- This is not a paragraph. -->
</section>
```

Soit *view* une vue du DOM qui remplace tous les éléments a, ins, del et map du document par leur contenu . Ensuite, en *vue* , pour chaque série de nœuds de

[contenu de phrasé](#) frères ininterrompus par d'autres types de contenu, dans un élément qui accepte un contenu autre que [le contenu de phrasé](#) ainsi que [le contenu de phrasé](#) , soit d'abord le premier nœud de la série, et soit le *dernier* le dernier nœud de la course. Pour chaque exécution de ce type composée d'au moins un nœud qui n'est ni [un contenu intégré](#) ni [un espace blanc inter-éléments](#) , un paragraphe existe dans le DOM d'origine juste avant *premier* à immédiatement après *dernier* . (Les paragraphes peuvent donc s'étendre sur [a](#), [ins](#), [del](#), et [map](#) .)

Les vérificateurs de conformité peuvent avertir les auteurs des cas où ils ont des paragraphes qui se chevauchent (cela peut se produire avec les éléments [object](#), [video](#), [audio](#), et [canvas](#), et indirectement via des éléments dans d'autres espaces de noms qui permettent au HTML d'être intégré davantage, comme [SVG](#)[svg](#) ou [MathML](#)[math](#) ).

Un [paragraphe](#) est également formé explicitement par [p](#) des éléments.

*L'élément peut être utilisé pour envelopper des paragraphes individuels lorsqu'il n'y aurait autrement aucun contenu autre que le contenu de phrase pour séparer les paragraphes les uns des autres.*

Dans l'exemple suivant, le lien couvre la moitié du premier paragraphe, tout le titre séparant les deux paragraphes et la moitié du deuxième paragraphe. Il chevauche les paragraphes et le titre.

```
<header>
  Welcome!
  <a href="about.html">
    This is home of...
  <h1>The Falcons!</h1>
  The Lockheed Martin multirole jet fighter aircraft!
</a>
  This page discusses the F-16 Fighting Falcon's innermost secrets.
</header>
```

Voici une autre façon de marquer cela, cette fois en montrant explicitement les paragraphes et en divisant l'élément de lien en trois :

```
<header>
  <p>Welcome! <a href="about.html">This is home of...</a></p>
  <h1><a href="about.html">The Falcons!</a></h1>
  <p><a href="about.html">The Lockheed Martin multirole jet
    fighter aircraft!</a> This page discusses the F-16 Fighting
    Falcon's innermost secrets.</p>
</header>
```

Il est possible que des paragraphes se chevauchent lors de l'utilisation de certains éléments qui définissent le contenu de secours. Par exemple, dans la section suivante :

```

<section>
  <h2>My Cats</h2>
  You can play with my cat simulator.
  <object data="cats.sim">
    To see the cat simulator, use one of the following links:
    <ul>
      <li><a href="cats.sim">Download simulator file</a>
      <li><a href="https://sims.example.com/watch?v=LYds5xY4INU">Use
online simulator</a>
    </ul>
    Alternatively, upgrade to the Mellblom Browser.
  </object>
  I'm quite proud of it.
</section>

```

Il y a cinq paragraphes :

1. Le paragraphe qui dit "Vous pouvez jouer avec mon simulateur de chat. *objet* J'en suis assez fier.", où *objet* est l' object élément.
2. Le paragraphe qui dit "Pour voir le simulateur de chat, utilisez l'un des liens suivants :".
3. Le paragraphe qui dit "Télécharger le fichier du simulateur".
4. Le paragraphe qui dit "Utiliser le simulateur en ligne".
5. Le paragraphe qui dit "Vous pouvez également effectuer une mise à niveau vers le navigateur Mellblom."

Le premier paragraphe est recouvert par les quatre autres. Un agent utilisateur qui prend en charge la ressource "cats.sim" n'affichera que le premier, mais un agent utilisateur qui affiche le repli affichera de manière confuse la première phrase du premier paragraphe comme si elle se trouvait dans le même paragraphe que le second, et affichera le dernier paragraphe comme s'il se trouvait au début de la deuxième phrase du premier paragraphe.

Pour éviter cette confusion, p des éléments explicites peuvent être utilisés. Par exemple:

```

<section>
  <h2>My Cats</h2>
  <p>You can play with my cat simulator.</p>
  <object data="cats.sim">
    <p>To see the cat simulator, use one of the following links:</p>
    <ul>
      <li><a href="cats.sim">Download simulator file</a>
      <li><a href="https://sims.example.com/watch?v=LYds5xY4INU">Use
online simulator</a>
    </ul>

```

```
</ul>

<p>Alternatively, upgrade to the Mellblom Browser.</p>

</object>

<p>I'm quite proud of it.</p>

</section>
```

### 3.2.6 Attributs globaux

MDN

Les attributs suivants sont communs et peuvent être spécifiés sur tous [les éléments HTML](#) (même ceux qui ne sont pas définis dans cette spécification) :

- [accesskey](#)
- [autocapitalize](#)
- [autofocus](#)
- [contenteditable](#)
- [dir](#)
- [draggable](#)
- [enterkeyhint](#)
- [hidden](#)
- [inert](#)
- [inputmode](#)
- [is](#)
- [itemid](#)
- [itemprop](#)
- [itemref](#)
- [itemscope](#)
- [itemtype](#)
- [lang](#)
- [nonce](#)
- [popover](#)
- [spellcheck](#)
- [style](#)
- [tabindex](#)
- [title](#)
- [translate](#)

Ces attributs ne sont définis par cette spécification que comme attributs pour [les éléments HTML](#) . Lorsque cette spécification fait référence à des éléments ayant ces attributs, les éléments provenant d'espaces de noms qui ne sont pas définis comme ayant ces attributs ne doivent pas être considérés comme étant des éléments ayant ces attributs.

Par exemple, dans le fragment XML suivant, l' *bogus* élément " " n'a pas d' [dir](#) attribut tel que défini dans cette spécification, bien qu'il ait un attribut avec le nom littéral

" dir". Ainsi, [la directionnalité](#) de l'élément le plus à l'intérieur [span](#) est ' [rtl](#) ', héritée de l' [div](#) élément indirectement via l' `bogus` élément " ".

```
<div xmlns="http://www.w3.org/1999/xhtml" dir="rtl">
  <bogus xmlns="https://example.net/ns" dir="ltr">
    <span xmlns="http://www.w3.org/1999/xhtml">
      </span>
    </bogus>
  </div>
```



DOM définit les exigences de l'agent utilisateur pour les `class` attributs `id`, et `slot` pour n'importe quel élément dans n'importe quel espace de noms. [\[DOM\]](#)

Les attributs `class`, `id` et `slot` peuvent être spécifiés sur tous [les éléments HTML](#) .

Lorsqu'il est spécifié sur [des éléments HTML](#) , l' `class` attribut doit avoir une valeur qui est un [ensemble de jetons séparés par des espaces](#) représentant les différentes classes auxquelles appartient l'élément.

*L'attribution de classes à un élément affecte la correspondance des classes dans les sélecteurs CSS, la `getElementsByClassName()` méthode dans le DOM et d'autres fonctionnalités similaires.*

*Il n'y a pas de restrictions supplémentaires sur les jetons que les auteurs peuvent utiliser dans l' `class` attribut, mais les auteurs sont encouragés à utiliser des valeurs qui décrivent la nature du contenu, plutôt que des valeurs qui décrivent la présentation souhaitée du contenu.*

Lorsqu'elle est spécifiée sur [les éléments HTML](#) , la `id` valeur de l'attribut doit être unique parmi tous les [ID de l'arborescence](#) de l'élément et doit contenir au moins un caractère. La valeur ne doit contenir aucun [espace ASCII](#) .

*L' `id` attribut spécifie [l'identifiant \(ID\) unique](#) de son élément .*

*Il n'y a pas d'autres restrictions sur la forme que peut prendre une pièce d'identité ; en particulier, les identifiants peuvent se composer uniquement de chiffres, commencer par un chiffre, commencer par un trait de soulignement, se composer uniquement de ponctuation, etc.*

*L'identifiant unique d'un élément peut être utilisé à diverses fins, notamment comme moyen de créer un lien vers des parties spécifiques d'un document à l'aide de [fragments](#) , comme moyen de cibler un élément lors de la création de scripts et comme moyen de styliser un élément spécifique à partir de CSS .*

Les identificateurs sont des chaînes opaques. Des significations particulières ne doivent pas être dérivées de la valeur de la [id](#) attribut.

Il n'y a pas d'exigences de conformité pour l' [slot](#) attribut spécifique aux [éléments HTML](#) .

*L' [slot](#) attribut est utilisé pour [assigner un emplacement](#) à un élément : un élément avec un [slot](#) attribut est [assigné](#) à l' [emplacement](#) créé par l' [slot](#) élément dont la valeur de l'attribut [name](#)[slot](#) correspond à la valeur de cet attribut — mais seulement si cet [slot](#) élément se trouve dans l' [arbre fantôme](#) dont la [racine host](#) a la [slot](#) valeur d'attribut correspondante.*

---

Pour permettre aux produits de technologie d'assistance d'exposer une interface plus fine que ce qui est autrement possible avec les éléments et attributs HTML, un ensemble d' [annotations pour les produits de technologie d'assistance](#) peut être spécifié (l'ARIA [role](#) et [aria-\\*](#) les attributs). [\[ARIA\]](#)

---

[Les attributs de contenu de gestionnaire d'événements](#) suivants peuvent être spécifiés sur n'importe quel [élément HTML](#) :

- [onauxclick](#)
- [onbeforeinput](#)
- [onbeforematch](#)
- [onbeforetoggle](#)
- [onblur](#)\*
- [oncancel](#)
- [oncanplay](#)
- [oncanplaythrough](#)
- [onchange](#)
- [onclick](#)
- [onclose](#)
- [oncontextlost](#)
- [oncontextmenu](#)
- [oncontextrestored](#)

- oncopy
- oncuechange
- oncut
- ondblclick
- ondrag
- ondragend
- ondragenter
- ondragleave
- ondragover
- ondragstart
- ondrop
- ondurationchange
- onemptied
- onended
- onerror\*
- onfocus\*
- onformdata
- oninput
- oninvalid
- onkeydown
- onkeypress
- onkeyup
- onload\*
- onloadeddata
- onloadedmetadata
- onloadstart
- onmousedown
- onmouseenter
- onmouseleave
- onmousemove
- onmouseout
- onmouseover
- onmouseup
- onpaste
- onpause
- onplay
- onplaying
- onprogress
- onratechange
- onreset
- onresize\*
- onscroll\*
- onscrollend\*
- onsecuritypolicyviolation
- onseeked
- onseeking
- onselect
- onslotchange
- onstalled
- onsubmit



- [onsuspend](#)
- [ontimeupdate](#)
- [ontoggle](#)
- [onvolumechange](#)
- [onwaiting](#)
- [onwheel](#)

*Les attributs marqués d'un astérisque ont une signification différente lorsqu'ils sont spécifiés sur [body](#) des éléments car ces éléments exposent [les gestionnaires d'événements](#) de l'[Window](#) objet avec les mêmes noms.*

*Bien que ces attributs s'appliquent à tous les éléments, ils ne sont pas utiles sur tous les éléments. Par exemple, seuls [les éléments multimédias](#) recevront un [volumechange](#) événement déclenché par l'agent utilisateur.*

[Les attributs de données personnalisés](#) (par exemple `data-foldername` ou `data-msgid`) peuvent être spécifiés sur n'importe quel [élément HTML](#), pour stocker des données personnalisées, l'état, les annotations et autres, spécifiques à la page.

Dans [les documents HTML](#), les éléments de l'[espace de noms HTML](#) peuvent avoir un `xmlns` attribut spécifié, si et seulement si, il a la valeur exacte "`http://www.w3.org/1999/xhtml`". Cela ne s'applique pas aux [documents XML](#).

*En HTML, l'`xmlns` attribut n'a absolument aucun effet. C'est essentiellement un talisman. Il est autorisé simplement à faciliter légèrement la migration vers et depuis XML. Lorsqu'il est analysé par un [analyseur HTML](#), l'attribut se retrouve dans aucun espace de noms, pas le "`http://www.w3.org/2000/xmlns/` espace de noms " " comme le font les attributs de déclaration d'espace de noms dans XML. En XML, un `xmlns` attribut fait partie du mécanisme de déclaration d'espace de noms et un élément ne peut pas avoir d'`xmlns` attribut dans aucun espace de noms spécifié.*

XML permet également l'utilisation de l'`xml:space` attribut dans l'[espace de noms XML](#) sur n'importe quel élément d'un [document XML](#). Cet attribut n'a aucun effet sur [les éléments HTML](#), car le comportement par défaut en HTML est de préserver les espaces.[\[XML\]](#)

*Il n'y a aucun moyen de sérialiser l' xml:space attribut sur les éléments HTML dans la text/html syntaxe.*

### 3.2.6.1 L' titleattribut



L' titleattribut représente des informations consultatives pour l'élément, telles qu'elles seraient appropriées pour une info-bulle. Sur un lien, cela peut être le titre ou une description de la ressource cible ; sur une image, il peut s'agir du crédit image ou d'une description de l'image ; sur un paragraphe, il pourrait s'agir d'une note de bas de page ou d'un commentaire sur le texte; sur une citation, il pourrait s'agir d'informations supplémentaires sur la source; sur un contenu interactif , il peut s'agir d'un libellé ou d'instructions pour l'utilisation de l'élément ; et ainsi de suite. La valeur est du texte.

*Il est actuellement déconseillé de s'appuyer sur l' titleattribut car de nombreux agents utilisateurs n'exposent pas l'attribut de manière accessible comme l'exige cette spécification (par exemple, en exigeant un dispositif de pointage tel qu'une souris pour faire apparaître une info-bulle, ce qui exclut les utilisateurs utilisant uniquement le clavier et utilisateurs tactiles uniquement, comme toute personne disposant d'un téléphone ou d'une tablette moderne).*

Si cet attribut est omis d'un élément, cela implique que l' titleattribut de l' élément HTML ancêtre le plus proche avec un titleensemble d'attributs est également pertinent pour cet élément. La définition de l'attribut remplace cela, indiquant explicitement que les informations consultatives de tous les ancêtres ne sont pas pertinentes pour cet élément. La définition de l'attribut sur la chaîne vide indique que l'élément n'a pas d'informations consultatives.

Si la titlevaleur de l'attribut contient des caractères U+000A LINE FEED (LF), le contenu est divisé en plusieurs lignes. Chaque caractère U+000A LINE FEED (LF) représente un saut de ligne.

La prudence est recommandée en ce qui concerne l'utilisation de retours à la ligne dans titleles attributs.

Par exemple, l'extrait de code suivant définit en fait l'expansion d'une abréviation *avec un saut de ligne* :

```
<p>My logs show that there was some interest in <abbr  
title="Hypertext  
Transport Protocol">HTTP</abbr> today.</p>
```

Certains éléments, tels que link, abbret input, définissent une sémantique supplémentaire pour l' titleattribut au-delà de la sémantique décrite ci-dessus.

Les **informations consultatives** d'un élément sont la valeur renvoyée par l'algorithme suivant, l'algorithme étant abandonné une fois qu'une valeur est renvoyée. Lorsque l'algorithme renvoie la chaîne vide, il n'y a pas d'informations consultatives.

1. Si l'élément a un [title](#)attribut, retourne sa valeur.
2. Si l'élément a un élément parent, renvoie les [informations consultatives](#) de l'élément parent .
3. Renvoie la chaîne vide.

Les agents utilisateurs doivent informer l'utilisateur lorsque des éléments ont [des informations consultatives](#) , sinon les informations ne seraient pas détectables.

---



L' [title](#)attribut IDL doit [refléter](#) l' [title](#)attribut content.

### 3.2.6.2 Les attributs [lang](#)et[xml:lang](#)



L' [lang](#)attribut (sans espace de noms) spécifie la langue principale pour le contenu de l'élément et pour tous les attributs de l'élément contenant du texte. Sa valeur doit être une balise de langue BCP 47 valide ou la chaîne vide. La définition de l'attribut sur la chaîne vide indique que la langue principale est inconnue. [\[BCP47\]](#)

L' [lang](#)attribut dans l' [espace de noms XML](#) est défini en XML. [\[XML\]](#)

Si ces attributs sont omis d'un élément, alors la langue de cet élément est la même que la langue de son élément parent, le cas échéant.

L' [lang](#)attribut dans aucun espace de noms peut être utilisé sur n'importe quel [élément HTML](#) .

L' [lang](#)attribut dans l' [espace de noms XML](#) peut être utilisé sur [des éléments HTML](#) dans [des documents XML](#) , ainsi que sur des éléments dans d'autres espaces de noms si les spécifications pertinentes le permettent (en particulier, MathML et

SVG autorisent la spécification [lang](#)d'attributs dans l' [espace de noms XML](#) sur leurs éléments). Si l' [lang](#)attribut dans aucun espace de noms et l' [lang](#)attribut dans l' [espace de noms XML](#) sont spécifiés sur le même élément, ils doivent avoir exactement la même valeur lorsqu'ils sont comparés d'une manière [ASCII insensible à la casse](#) .

Les auteurs ne doivent pas utiliser l' [lang](#)attribut dans l' [espace de noms XML](#) sur [les éléments HTML](#) dans [les documents HTML](#) . Pour faciliter la migration vers et depuis XML, les auteurs peuvent spécifier un attribut dans aucun espace de noms sans préfixe et avec le nom local littéral " " sur les éléments `xml:lang`HTML [dans les documents HTML](#) , mais ces attributs ne doivent être spécifiés que si un [lang](#)attribut dans aucun espace de noms est également spécifié, et les deux attributs doivent avoir la même valeur lorsqu'ils sont comparés d'une manière [ASCII insensible à la casse](#) .

*L'attribut dans aucun espace de noms sans préfixe et avec le nom local littéral " xml:lang " n'a aucun effet sur le traitement du langage.*

---

Pour déterminer la **langue** d'un nœud, les agents utilisateurs doivent rechercher l'élément ancêtre le plus proche (y compris l'élément lui-même si le nœud est un élément) qui a un [lang](#)attribut dans l' [ensemble d' espace de noms XML](#) ou qui est un [élément HTML](#) et qui a un [lang](#)attribut in no namespace ensemble. Cet attribut spécifie la langue du nœud (quelle que soit sa valeur).

Si à la fois l' [lang](#)attribut dans aucun espace de noms et l' [lang](#)attribut dans l' [espace de noms XML](#) sont définis sur un élément, les agents utilisateurs doivent utiliser l' [lang](#)attribut dans l' [espace de noms XML](#) et l' [lang](#)attribut dans aucun espace de noms doit être [ignoré](#) dans le but de déterminer la langue de l'élément.

Si les [ancêtres inclusifs](#) du nœud n'ont aucun ensemble d'attributs, mais qu'il existe un ensemble [de langues par défaut pragma-set](#) , alors il s'agit de la langue du nœud. S'il n'y a pas [de jeu de langues par défaut pragma-set](#) , les informations de langue d'un protocole de niveau supérieur (tel que HTTP), le cas échéant, doivent être utilisées comme langue de secours finale à la place. En l'absence de telles informations de langue, et dans les cas où le protocole de niveau supérieur signale plusieurs langues, la langue du nœud est inconnue et la balise de langue correspondante est la chaîne vide.

Si la valeur résultante n'est pas une étiquette de langue reconnue, elle doit être traitée comme une langue inconnue ayant l'étiquette de langue donnée, distincte de toutes les autres langues. Aux fins d'aller-retour ou de communication avec d'autres services qui attendent des balises de langue, les agents utilisateurs doivent transmettre des balises de langue inconnues non modifiées et étiquetées comme

étant des balises de langue BCP 47, afin que les services ultérieurs n'interprètent pas les données comme un autre type de langue. description. [\[BCP47\]](#)

Ainsi, par exemple, un élément avec `lang="xyzzzy"` serait mis en correspondance par le sélecteur `:lang(xyzzzy)` (par exemple en CSS), mais il ne serait pas mis en correspondance par `:lang(abcde)`, même si les deux sont également invalides. De même, si un navigateur Web et un lecteur d'écran travaillant à l'unisson communiquaient sur la langue de l'élément, le navigateur indiquerait au lecteur d'écran que la langue était "xyzzzy", même s'il savait qu'elle n'était pas valide, juste au cas où le lecteur d'écran en fait pris en charge une langue avec cette balise après tout. Même si le lecteur d'écran prenait en charge à la fois BCP 47 et une autre syntaxe pour encoder les noms de langue, et dans cette autre syntaxe, la chaîne "xyzzzy" était un moyen de désigner la langue biélorusse, ce serait *incorrect* pour que le lecteur d'écran commence alors à traiter le texte comme biélorusse, car "xyzzzy" n'est pas la façon dont le biélorusse est décrit dans les codes BCP 47 (BCP 47 utilise le code "be" pour le biélorusse).

Si la valeur résultante est la chaîne vide, alors elle doit être interprétée comme signifiant que la langue du nœud est explicitement inconnue.

---

Les agents utilisateurs peuvent utiliser la langue de l'élément pour déterminer le traitement ou le rendu approprié (par exemple dans la sélection des polices ou des prononciations appropriées, pour la sélection du dictionnaire ou pour les interfaces utilisateur des contrôles de formulaire tels que les sélecteurs de date).

---



L' **lang**attribut IDL doit [refléter](#) l' [lang](#)attribut content dans aucun espace de noms.

### 3.2.6.3 L' [translate](#)attribut



L' **translate** attribut est un [attribut énuméré](#) qui est utilisé pour spécifier si les valeurs d'attribut d'un élément et les valeurs de ses **Text**nœuds enfants doivent être traduites lorsque la page est localisée, ou s'il faut les laisser inchangées.

Les mots clés de l'attribut sont la chaîne vide, **yes** et **no**. La chaîne vide et le **yes** mot-clé correspondent à l'état *oui*. Le **no** mot-clé correspond à l'état *no*. De plus, il existe un troisième état, l'état *d'héritage*, qui est la [valeur manquante default](#) et la [valeur invalide default](#).

Chaque élément (même les éléments non-HTML) a un **mode de traduction**, qui est soit dans l'état [de traduction activé](#), soit dans l'état [de non-traduction](#). Si l'attribut d'un [élément HTML](#) **translate** est à l'état *oui*, alors le [mode de traduction](#) de l'élément est à l'état [translate-enabled](#); sinon, si l' **translate** attribut de l'élément est dans l'état *no*, alors le [mode de traduction](#) de l'élément est dans l'état [no-translate](#). Sinon, soit l' **translate** attribut de l'élément est dans l'état *d'héritage*, soit l'élément n'est pas un [élément HTML](#) et n'a donc pas d' **translate** attribut; dans les deux cas, le [mode de traduction](#) de l'élément est dans le même état que celui de son [élément parent](#), le cas échéant, ou dans l'état [de traduction activé](#), si l' [élément parent](#) de l'élément est nul.

Lorsqu'un élément est dans l'état **translate-enabled**, les [attributs traduisibles](#) de l'élément et les valeurs de ses **Text**nœuds enfants doivent être traduits lorsque la page est localisée.

Lorsqu'un élément est dans l'état **no-translate**, les valeurs d'attribut de l'élément et les valeurs de ses **Text**nœuds enfants doivent être laissées telles quelles lorsque la page est localisée, par exemple parce que l'élément contient le nom d'une personne ou le nom d'un programme informatique.

Les attributs suivants sont **des attributs traduisibles** :

- [abbr](#) sur [th](#) les éléments
- [alt](#) sur les éléments [area](#), [img](#) et [input](#)
- [content](#) sur [meta](#) les éléments, si l' [name](#) attribut spécifie un nom de métadonnées dont la valeur est connue pour être traduisible
- [download](#) sur [a](#) et [area](#) éléments
- [label](#) sur les éléments [optgroup](#), [option](#) et [track](#)
- [lang](#) sur [les éléments HTML](#); doit être "traduit" pour correspondre à la langue utilisée dans la traduction
- [placeholder](#) sur [input](#) et [textarea](#) éléments
- [srcdoc](#) sur [iframe](#) les éléments; doit être analysé et traité de manière récursive
- [style](#) sur [les éléments HTML](#); doit être analysé et traité de manière récursive (par exemple pour les valeurs des propriétés ['content'](#))
- [title](#) sur tous [les éléments HTML](#)
- [value](#) sur [input](#) les éléments avec un [type](#) attribut à l'état [Button](#) ou à l'état [Reset Button](#)

D'autres spécifications peuvent définir d'autres attributs qui sont également [des attributs traduisibles](#) . Par exemple, *ARIA* définirait l' [aria-label](#) attribut comme traduisible.

---

L' **translate** attribut IDL doit, à l'obtention, retourner true si le [mode de traduction](#) de l'élément est [translate-enabled](#) , et false sinon. Lors de la définition, il doit définir la valeur de l'attribut de contenu sur " yes " si la nouvelle valeur est vraie, et définir la valeur de l'attribut de contenu sur " no " sinon.

Dans cet exemple, tout le contenu du document doit être traduit lorsque la page est localisée, à l'exception de l'exemple d'entrée au clavier et de l'exemple de sortie du programme :

```
<!DOCTYPE HTML>
<html lang=en> <!-- default on the document element is
translate=yes -->
  <head>
    <title>The Bee Game</title> <!-- implied translate=yes inherited
from ancestors -->
  </head>
  <body>
    <p>The Bee Game is a text adventure game in English.</p>
    <p>When the game launches, the first thing you should do is type
    <kbd translate=no>eat honey</kbd>. The game will respond
with:</p>
    <pre><samp translate=no>Yum yum! That was some good
honey!</samp></pre>
  </body>
</html>
```

#### 3.2.6.4 L' **dir** attribut



L' **dir** attribut spécifie la directionnalité du texte de l'élément. L'attribut est un [attribut énuméré](#) avec les mots clés et les états suivants :

Le **ltr** mot-clé, qui correspond à l' état *ltr*

Indique que le contenu de l'élément est un texte explicitement isolé de gauche à droite.

**Le rtl mot-clé, qui correspond à l' état rtl**

Indique que le contenu de l'élément est un texte explicitement isolé de droite à gauche.

**Le auto mot-clé, qui correspond à l' état automatique**

Indique que le contenu de l'élément est un texte explicitement isolé de manière directionnelle, mais que la direction doit être déterminée par programme à l'aide du contenu de l'élément (comme décrit ci-dessous).

*L'heuristique utilisée par cet état est très grossière (elle ne regarde que le premier caractère avec une forte directionnalité, d'une manière analogue à la détermination du niveau de paragraphe dans l'algorithme bidirectionnel). Les auteurs sont priés de n'utiliser cette valeur qu'en dernier recours lorsque la direction du texte est vraiment inconnue et qu'aucune meilleure heuristique côté serveur ne peut être appliquée. [\[BIDI\]](#)  
Pour les éléments textarea et pre, l'heuristique est appliquée au niveau du paragraphe.*

L'attribut n'a pas [de valeur par défaut invalide](#) ni [de valeur par défaut manquante](#) .

---

**La directionnalité** d'un élément (n'importe quel élément, pas seulement un [élément HTML](#) ) est soit ' ltr ' soit ' rtl ', et est déterminée selon la première série d'étapes appropriées de la liste suivante :

Si l' dir attribut de l'élément est dans l' état ltr

Si l'élément est un [élément de document](#) et que l' dir attribut n'est pas dans un état défini (c'est-à-dire qu'il n'est pas présent ou a une valeur invalide)

Si l'élément est un input élément dont type l'attribut est dans l' état [Téléphone](#) , et que l' dir attribut n'est pas dans un état défini (c'est-à-dire qu'il n'est pas présent ou a une valeur invalide)

[La directionnalité](#) de l'élément est ' ltr '.

Si l' dir attribut de l'élément est à l' état rtl

[La directionnalité](#) de l'élément est ' rtl '.

Si l'élément est un input élément dont type l'attribut est à

l'état [Text](#) , [Search](#) , [Telephone](#) , [URL](#) ou [Email](#) et que l' dir attribut est à l' état auto



Si l'élément est un **textarea**élément et que l' **dir** attribut est dans l' état **automatique**

Si la **valeur** de l'élément contient un caractère de type de caractère bidirectionnel AL ou R, et qu'il n'y a aucun caractère de type de caractère bidirectionnel L avant lui dans la **valeur** de l'élément , alors **la directionnalité** de l'élément est ' **rtl** '. **[BIDI]**

Sinon, si la **valeur** de l'élément n'est pas la chaîne vide, ou si l'élément est un **élément de document** , **la directionnalité** de l'élément est ' **ltr** '.

Sinon, **la directionnalité** de l'élément est la même que la **directionnalité** de l'élément parent de l'élément .

Si l' **dir**attribut de l'élément est dans l' **auto** état

Si l'élément est un **bdi**élément et que l' **dir** attribut n'est pas dans un état défini (c'est-à-dire qu'il n'est pas présent ou a une valeur invalide)

Recherchez le premier caractère dans **l'arborescence** qui correspond aux critères suivants :

- Le caractère provient d'un **Text**nœud qui est un descendant de l'élément dont **la directionnalité** est déterminée.
- Le caractère est de type bidirectionnel L, AL ou R. **[BIDI]**
- Le personnage n'est pas dans un **Text**nœud qui a un élément ancêtre qui est un descendant de l'élément dont **la directionnalité** est déterminée et qui est soit :
  - Un**bdi**élément.
  - Un **script**élément.
  - Un **style**élément.
  - Un **textarea**élément.
  - Un élément avec un **dir**attribut dans un état défini.

Si un tel caractère est trouvé et qu'il est de type de caractère bidirectionnel AL ou R, **la directionnalité** de l'élément est ' **rtl** '.

Si un tel caractère est trouvé et qu'il est de type de caractère bidirectionnel L, **la directionnalité** de l'élément est ' **ltr** '.

Sinon, si l'élément est un **élément de document** , **la directionnalité** de l'élément est ' **ltr** '.

Sinon, **la directionnalité** de l'élément est la même que la **directionnalité** de l'élément parent de l'élément .

Si l'élément a un élément parent et que l' **dir**attribut n'est pas dans un état défini (c'est-à-dire qu'il n'est pas présent ou a une valeur invalide)

La directionnalité de l'élément est la même que la directionnalité de l'élément parent de l'élément .

*Étant donné que l' dirattribut n'est défini que pour les éléments HTML , il ne peut pas être présent sur les éléments d'autres espaces de noms. Ainsi, les éléments d'autres espaces de noms héritent toujours leur directionnalité de leur élément parent, ou, s'ils n'en ont pas, la valeur par défaut est ' ltr ' .  
Cet attribut a des exigences de rendu impliquant l'algorithme bidirectionnel .*

---

La **directionnalité d'un attribut** d'un élément HTML , qui est utilisée lorsque le texte de cet attribut doit être inclus dans le rendu d'une certaine manière, est déterminée selon la première série d'étapes appropriées de la liste suivante :

Si l'attribut est un attribut capable de directionnalité et que l' dirattribut de l'élément est à l' état automatique

Trouver le premier caractère (dans l'ordre logique) de la valeur de l'attribut qui est de type de caractère bidirectionnel L, AL ou R. [BIDI]

Si un tel caractère est trouvé et qu'il est de type de caractère bidirectionnel AL ou R, la directionnalité de l'attribut est ' rtl ' .

Sinon, la directionnalité de l'attribut est ' ltr ' .

**Sinon**

La directionnalité de l'attribut est la même que la directionnalité de l'élément .

Les attributs suivants sont **des attributs compatibles avec la direction** :

- abbr sur th les éléments
- alt sur les éléments area, img et input
- content sur meta les éléments, si l' name attribut spécifie un nom de métadonnées dont la valeur est principalement destinée à être lisible par l'homme plutôt que lisible par la machine
- label sur les éléments optgroup, option et track
- placeholder sur input et textarea éléments
- title sur tous les éléments HTML

---

```
document.dir [ = value ]
```

Renvoie [la<sub>html</sub>](#) valeur de l'attribut de l' [élément<sub>dir</sub>](#) , le cas échéant.

Peut être défini sur " ltr", " rtl" ou " auto" pour remplacer [la<sub>html</sub><sub>dir</sub>](#) valeur de l'attribut de l' [élément](#) .

S'il n'y a pas [html<sub>d'élément</sub>](#) , renvoie la chaîne vide et ignore les nouvelles valeurs.

✓ MDN

L' [dir](#)attribut IDL d'un élément doit [réfléter](#) l' [dir](#)attribut content de cet élément, [limité aux seules valeurs connues](#) .

✓ MDN

L' [dir](#)attribut IDL sur [Document](#)les objets doit [réfléter](#) l' [dir](#)attribut content de l' [htmlélément](#) , le cas échéant, [limité aux seules valeurs connues](#) . S'il n'y a pas un tel élément, alors l'attribut doit renvoyer la chaîne vide et ne rien faire lors de la définition.

*Les auteurs sont fortement encouragés à utiliser l' [dir](#) attribut pour indiquer la direction du texte plutôt qu'à utiliser CSS, car ainsi leurs documents continueront à s'afficher correctement même en l'absence de CSS (par exemple, tel qu'interprété par les moteurs de recherche).*

Ce fragment de balisage provient d'une conversation par messagerie instantanée.

```
<p dir=auto class="u1"><b><bdi>Student</bdi>:</b> How do you write  
"What's your name?" in Arabic?</p>  
<p dir=auto class="u2"><b><bdi>Teacher</bdi>:</b> ما اسمك؟</p>  
<p dir=auto class="u1"><b><bdi>Student</bdi>:</b> Thanks.</p>  
<p dir=auto class="u2"><b><bdi>Teacher</bdi>:</b> That's written  
"شكراً".</p>  
<p dir=auto class="u2"><b><bdi>Teacher</bdi>:</b> Do you know how  
to write "Please"?</p>  
<p dir=auto class="u1"><b><bdi>Student</bdi>:</b> "من فضلك",  
right?</p>
```

Étant donné une feuille de style appropriée et les styles d'alignement par défaut pour l' [p](#)élément, à savoir pour aligner le texte sur le *bord de début* du paragraphe, le rendu résultant pourrait être le suivant :

**Student:** How do you write "What's your name?" in Arabic?

**Teacher:** ما اسمك؟

**Student:** Thanks.

**Teacher:** That's written "شكراً".

**Teacher:** Do you know how to write "Please"?

**Student:** "من فضلك", right?

Comme indiqué précédemment, la [auto](#)valeur n'est pas une panacée. Le dernier paragraphe de cet exemple est interprété à tort comme étant du texte de droite à gauche, car il commence par un caractère arabe, ce qui provoque le "droit ?" être à gauche du texte arabe.

### 3.2.6.5 L' [style](#)attribut



Tous [les éléments HTML](#) peuvent avoir l' [style](#)attribut content défini. Il s'agit d'un [attribut de style](#) tel que défini par *CSS Style Attributes* . [\[CSSATTR\]](#)

Dans les agents utilisateurs prenant en charge CSS, la valeur de l'attribut doit être analysée lorsque l'attribut est ajouté ou que sa valeur est modifiée, conformément aux règles données pour les [attributs de style](#) . [\[CSSATTR\]](#)

Toutefois, si le [comportement en ligne de l'élément doit-il être bloqué par la politique de sécurité du contenu ?](#) L'algorithme renvoie "Blocked" lorsqu'il est exécuté sur l' [élément](#) de l'attribut , " `style attribute` " et la valeur de l'attribut, alors les règles de style définies dans la valeur de l'attribut ne doivent pas être appliquées à l' [élément](#) . [\[CSP\]](#)

Les documents qui utilisent [style](#)des attributs sur l'un de leurs éléments doivent toujours être compréhensibles et utilisables si ces attributs ont été supprimés.

*En particulier, l'utilisation de l' [style](#)attribut pour masquer et afficher du contenu, ou pour transmettre une signification qui n'est pas incluse dans le document, est non conforme. (Pour masquer et afficher le contenu, utilisez l' [hidden](#) attribut.)*

---

#### `element.style`

Renvoie un [CSSStyleDeclaration](#)objet pour l' [style](#)attribut de l'élément.

L' [style](#)attribut IDL est défini dans *CSS Object Model* . [\[CSSOM\]](#)

Dans l'exemple suivant, les mots qui font référence aux couleurs sont marqués à l'aide de l' [span](#)élément et de l' [style](#)attribut pour que ces mots apparaissent dans les couleurs appropriées dans les médias visuels.

```
<p>My sweat suit is <span style="color: green; background: transparent">green</span> and my eyes are <span style="color: blue;
```

```
background: transparent">blue</span>.</p>
```

### 3.2.6.6 Incorporation de données personnalisées non visibles avec les data-\*attributs



Un **attribut de données personnalisé** est un attribut dans aucun espace de noms dont le nom commence par la chaîne " **data-**", comporte au moins un caractère après le trait d'union, est [compatible XML](#) et ne contient pas [d'alphas supérieurs ASCII](#) .

*Tous les noms d'attributs des [éléments HTML](#) dans [les documents HTML](#) sont automatiquement mis en minuscules ASCII, de sorte que la restriction sur les lettres majuscules ASCII n'affecte pas ces documents.*

[Les attributs de données personnalisés](#) sont destinés à stocker des données personnalisées, des états, des annotations et similaires, privés à la page ou à l'application, pour lesquels il n'y a plus d'attributs ou d'éléments appropriés.

Ces attributs ne sont pas destinés à être utilisés par un logiciel qui n'est pas connu des administrateurs du site qui utilise les attributs. Pour les extensions génériques qui doivent être utilisées par plusieurs outils indépendants, soit cette spécification doit être étendue pour fournir explicitement la fonctionnalité, soit une technologie telle que [les microdonnées](#) doit être utilisée (avec un vocabulaire normalisé).

Par exemple, un site sur la musique pourrait annoter les éléments de liste représentant les pistes d'un album avec des attributs de données personnalisés contenant la longueur de chaque piste. Ces informations pourraient ensuite être utilisées par le site lui-même pour permettre à l'utilisateur de trier la liste par longueur de piste ou de filtrer la liste pour les pistes d'une certaine longueur.

```
<ol>
  <li data-length="2m11s">Beyond The Sea</li>
  ...
</ol>
```

Il serait toutefois inapproprié que l'utilisateur utilise un logiciel générique non associé à ce site de musique pour rechercher des pistes d'une certaine longueur en consultant ces données.

En effet, ces attributs sont destinés à être utilisés par les propres scripts du site et ne constituent pas un mécanisme d'extension générique pour les métadonnées utilisables publiquement.

De même, un auteur de page pourrait écrire un balisage qui fournit des informations pour un outil de traduction qu'il a l'intention d'utiliser :

```
<p>The third <span data-mytrans-de="Anspruch">claim</span> covers  
the case of <span  
translate="no">HTML</span> markup.</p>
```

Dans cet exemple, l'attribut `data-mytrans-de` " " donne un texte spécifique que le produit MyTrans doit utiliser lors de la traduction de l'expression "revendication" en allemand. Cependant, l'attribut standard `translate` est utilisé pour lui dire que dans toutes les langues, "HTML" doit rester inchangé. Lorsqu'un attribut standard est disponible, il n'est pas nécessaire d'utiliser un [attribut de données personnalisé](#).

Dans cet exemple, des attributs de données personnalisés sont utilisés pour stocker le résultat d'une détection de fonctionnalité pour `PaymentRequest`, qui pourrait être utilisé dans CSS pour styliser différemment une page de paiement.

```
<script>  
  if ('PaymentRequest' in window) {  
    document.documentElement.dataset.hasPaymentRequest = '';  
  }  
</script>
```

Ici, l'attribut `data-has-payment-request` est effectivement utilisé comme un [attribut booléen](#) ; il suffit de vérifier la présence de l'attribut. Cependant, si l'auteur le souhaite, il pourrait être rempli ultérieurement avec une certaine valeur, peut-être pour indiquer une fonctionnalité limitée de la fonctionnalité.

Chaque [élément HTML](#) peut avoir n'importe quel nombre d' [attributs de données personnalisés](#) spécifiés, avec n'importe quelle valeur.

Les auteurs doivent soigneusement concevoir de telles extensions afin que lorsque les attributs sont ignorés et que tout CSS associé soit supprimé, la page soit toujours utilisable.

Les agents utilisateurs ne doivent dériver aucun comportement d'implémentation de ces attributs ou valeurs. Les spécifications destinées aux agents utilisateurs ne doivent pas définir ces attributs pour qu'ils aient des valeurs significatives.

Les bibliothèques JavaScript peuvent utiliser les [attributs de données personnalisés](#), car ils sont considérés comme faisant partie de la page sur laquelle ils sont utilisés. Les auteurs de bibliothèques qui sont réutilisées par de nombreux auteurs sont encouragés à inclure leur nom dans les noms d'attributs, afin de réduire le risque de conflits. Lorsque cela a du sens, les auteurs de bibliothèques sont également encouragés à personnaliser le nom exact utilisé dans les noms d'attributs, afin que les bibliothèques dont les auteurs ont choisi le même nom sans le savoir puissent être utilisées sur la même page, et afin que plusieurs versions d'une bibliothèque particulière puissent être utilisées sur la même page même lorsque ces versions ne sont pas mutuellement compatibles.

Par exemple, une bibliothèque appelée "DoQuery" pourrait utiliser des noms d'attributs comme `data-doquery-range`, et une bibliothèque appelée "jJo" pourrait utiliser des noms d'attributs comme `data-jjo-range`. La bibliothèque jJo pourrait également fournir une API pour définir le préfixe à utiliser (par exemple `J.setDataPrefix('j2')`), faire en sorte que les attributs aient des noms comme `data-j2-range`).

---

#### `element.dataset`

✓

Renvoie un [DOMStringMap](#) objet pour les `data-*` attributs de l'élément. Les noms avec trait d'union deviennent en forme de chameau. Par exemple, `data-foo-bar=""` devient `element.dataset.fooBar`.

L' `dataset` attribut IDL fournit des accesseurs pratiques pour tous les `data-*` attributs d'un élément. A l'obtention, l' `dataset` attribut IDL doit retourner un [DOMStringMap](#) dont l'élément associé est cet élément.

L' [DOMStringMap](#) interface est utilisée pour l' `dataset` attribut. Chacun [DOMStringMap](#) a un **élément associé** .

```
[Exposed=Window,  
  
LegacyOverrideBuiltIns]  
  
interface DOMStringMap {  
  
    getter DOMString (DOMString name);  
  
    [CEReactions] setter undefined (DOMString name, DOMString  
value);  
  
    [CEReactions] deleter undefined (DOMString name);  
  
};
```

Pour **obtenir** [DOMStringMap](#) les paires nom-valeur de `a` , exécutez l'algorithme suivant :

1. Soit *list* une liste vide de paires nom-valeur.

2. Pour chaque attribut de contenu sur l' [DOMStringMap](#)élément [associé](#) de dont les cinq premiers caractères sont la chaîne " data-" et dont les caractères restants (le cas échéant) n'incluent aucun [alpha supérieur ASCII](#) , dans l'ordre dans lequel ces attributs sont répertoriés dans la [liste d'attributs](#) de l'élément , ajoutez une paire nom-valeur à *liste* dont le nom est le nom de l'attribut avec les cinq premiers caractères supprimés et dont la valeur est la valeur de l'attribut.
3. Pour chaque nom de *la liste* , pour chaque caractère U+002D HYPHEN-MINUS (-) dans le nom qui est suivi d'un [alpha inférieur ASCII](#) , supprimez le caractère U+002D HYPHEN-MINUS (-) et remplacez le caractère qui le suivait par le même caractère [converti en majuscule ASCII](#) .
4. *Liste* de retour .

Les [noms de propriété pris en charge](#) sur un [DOMStringMap](#)objet à tout instant sont les noms de chaque paire renvoyée par [l'obtention des DOMStringMappaires nom-valeur de](#) à cet instant, dans l'ordre renvoyé.

Pour [déterminer la valeur d'un](#) *nom* de propriété nommé pour a [DOMStringMap](#), renvoyez le composant valeur de la paire nom-valeur dont le composant nom est *name* dans la liste renvoyée par [l'obtention des DOMStringMappaires nom-valeur de](#) .

Pour [définir la valeur d'une nouvelle propriété nommée](#) ou [définir la valeur d'une propriété nommée existante](#) pour un [DOMStringMap](#), étant donné un nom de propriété *name* et une nouvelle valeur *value* , exécutez les étapes suivantes :

1. Si *name* contient un caractère U+002D HYPHEN-MINUS (-) suivi d'un [alpha ASCII inférieur](#) , lancez un "[SyntaxError](#)" [DOMException](#) .
2. Pour chaque [alpha supérieur ASCII](#) dans *name* , insérez un caractère U+002D HYPHEN-MINUS (-) avant le caractère et remplacez le caractère par le même caractère [converti en ASCII minuscule](#) .
3. Insérez la chaîne `data-` devant *name* .
4. Si *le nom* ne correspond pas à la [Name](#)production XML, lancez un "[InvalidCharacterError](#)" [DOMException](#) .
5. [Définissez une valeur d'attribut](#) pour l' [élément associé](#)[DOMStringMap](#) à en utilisant *name* et *value* .

Pour [supprimer un](#) *nom* de propriété nommé existant pour un [DOMStringMap](#), exécutez les étapes suivantes :

1. Pour chaque [alpha supérieur ASCII](#) dans *name* , insérez un caractère U+002D HYPHEN-MINUS (-) avant le caractère et remplacez le caractère par le même caractère [converti en ASCII minuscule](#) .



2. Insérez la chaîne `data-` devant *name* .
3. Supprime un attribut par nom *prénom* et l' élément associé `DOMStringMap` à .

*Cet algorithme ne sera invoqué par Web IDL que pour les noms donnés par l'algorithme précédent pour obtenir les `DOMStringMap` paires nom-valeur de . [WEBIDL]*

Si une page Web voulait qu'un élément représente un vaisseau spatial, par exemple dans le cadre d'un jeu, elle devrait utiliser l' class attribut avec data-\* les attributs :

```
<div class="spaceship" data-ship-id="92432"
    data-weapons="laser 2" data-shields="50%"
    data-x="30" data-y="10" data-z="90">
  <button class="fire"
    onclick="spaceships[this.parentNode.dataset.shipId].fire()">
    Fire
  </button>
</div>
```

Remarquez comment le nom d'attribut avec trait d'union devient en casse camel dans l'API.

Étant donné le fragment suivant et les éléments avec des constructions similaires :

```

```

...on pourrait imaginer une fonction `splashDamage()` qui prend quelques arguments, dont le premier est l'élément à traiter :

```
function splashDamage(node, x, y, damage) {
  if (node.classList.contains('tower') && // checking the 'class'
      attribute
      node.dataset.x == x && // reading the 'data-x' attribute
      node.dataset.y == y) { // reading the 'data-y' attribute
    var hp = parseInt(node.dataset.hp); // reading the 'data-hp'
    attribute
    hp = hp - damage;
    if (hp < 0) {
      hp = 0;
      node.dataset.ai = 'dead'; // setting the 'data-ai' attribute
      delete node.dataset.ability; // removing the 'data-ability'
      attribute
    }
  }
}
```

```
node.dataset.hp = hp; // setting the 'data-hp' attribute
}
}
```

### 3.2.7 Les propriétés innerText et outerText



`element.innerText [ = value ]`

Renvoie le contenu textuel de l'élément "tel que rendu".

Peut être défini pour remplacer les enfants de l'élément par la valeur donnée, mais avec des sauts de ligne convertis en br éléments.

`element.outerText [ = value ]`

Renvoie le contenu textuel de l'élément "tel que rendu".

Peut être défini pour remplacer l'élément par la valeur donnée, mais avec des sauts de ligne convertis en br éléments.



Les étapes innerText et outerText getter sont :

1. Si ce n'est pas rendu ou si l'agent utilisateur est un agent utilisateur non-CSS, alors renvoyez le contenu textuel descendant de this .

*Cette étape peut produire des résultats surprenants, car lorsque le innerText getter est appelé sur un élément qui n'est pas rendu , son contenu textuel est renvoyé, mais lorsqu'il est accédé sur un élément qui est rendu , tous ses enfants qui ne sont pas rendus ont leur contenu textuel ignoré.*

2. Soit *résultats* une nouvelle liste vide .
3. Pour chaque nœud nœud *enfant* de this :

1. Soit *current* la liste résultant de l'exécution des étapes de collecte de texte rendu avec *node* . Chaque élément dans *les résultats* sera soit une chaîne , soit un entier positif (un *nombre de sauts de ligne requis* ).

*Intuitivement, un élément de nombre de sauts de ligne requis signifie qu'un certain nombre de sauts de ligne apparaissent à ce point, mais ils peuvent être réduits avec les sauts de ligne induits par les éléments de nombre de sauts de ligne requis adjacents , rappelant la réduction des marges CSS.*

2. Pour chaque élément *élément* dans *courant* , ajouter *élément* aux *résultats* .

4. [Supprimez](#) tous les éléments des *résultats* qui sont la chaîne vide.
5. [Supprimez](#) toutes les séries d'éléments *de nombre de sauts de ligne requis* consécutifs au début ou à la fin des *résultats* .
6. [Remplacez](#) chaque série restante d'éléments *de nombre de sauts de ligne requis* consécutifs par une chaîne composée d'autant de points de code LF U + 000A que le maximum des valeurs dans les éléments *de nombre de sauts de ligne requis* .
7. Renvoie la concaténation des éléments de chaîne dans *les résultats* .

Les **étapes de collecte de texte rendu** , étant donné un [nœud node](#) , sont les suivantes :

1. Supposons que *les éléments* soient le résultat de l'exécution des [étapes de collecte de texte rendu](#) avec chaque nœud enfant du *nœud* dans [l'ordre de l'arborescence](#) , puis de la concaténation des résultats en une seule [liste](#) .
2. Si [la valeur calculée](#) de ['visibility'](#) par *node* n'est pas 'visible', alors retourne *items* .
3. Si *node* n'est pas [rendu](#) , retournez *items* . Pour les besoins de cette étape, les éléments suivants doivent agir comme décrit si la [valeur calculée](#) de la propriété ['display'](#) n'est pas 'none' :
  - [select](#) [les éléments ont une boîte CSS](#) en ligne non remplacée associée dont les boîtes enfants incluent uniquement celles des nœuds enfants de [optgroup](#) et de l'élément ; [option](#)
  - [optgroup](#) [les éléments ont une boîte CSS](#) associée non remplacée au niveau du bloc dont les boîtes enfants incluent uniquement celles des [option](#) nœuds enfants de l'élément ; et
  - [option](#) [L'élément a une boîte CSS](#) de niveau bloc non remplacée associée dont les boîtes enfants sont comme d'habitude pour [les boîtes CSS](#) de niveau bloc non remplacées .

[les éléments peuvent être non vides en raison de 'display:contents'.](#)
4. Si *node* est un [Text](#) nœud, alors pour chaque zone de texte CSS produite par *node* , dans l'ordre du contenu, calculez le texte de la zone après application des règles de traitement CSS ['white-space'](#) et des règles ['text-transform'](#) , définissez les *éléments* sur [liste](#) des chaînes résultantes et renvoie *items* . Les règles de traitement [des "espaces blancs"](#) CSS sont légèrement modifiées : les espaces réductibles à la fin des lignes sont toujours réduits, mais ils ne sont supprimés que si la ligne est la dernière ligne du bloc, ou si elle se termine par un élément [br](#) . Les traits d'union souples doivent être conservés. [\[CSSTEXT\]](#)
5. Si *node* est un [br](#) élément, alors [ajoutez](#) une chaîne contenant un seul point de code U+000A LF à *items* .

6. Si [la valeur calculée](#) de `'display'` du *nœud* est `'table-cell'` et que [la boîte CSS](#) du *nœud* n'est pas la dernière boîte `'table-row'` englobante, alors [ajoutez](#) une chaîne contenant un seul U+0009 Point de code TAB vers *les articles*.
7. Si [la valeur calculée](#) de `'display'` du *nœud* est `'table-row'` et que [la case CSS](#) du *nœud* n'est pas la dernière case `'table-row'` de la case `'table'` ancêtre la plus proche, alors [ajoutez](#) une chaîne contenant un seul U +000A LF point de code aux *éléments*.
8. Si *node* est un *p*élément, [ajoutez](#) 2 (un *nombre de sauts de ligne requis*) au début et à la fin de *items*.
9. Si la [valeur 'display'](#) utilisée *par node* est [au niveau du bloc](#) ou `'table-caption'`, alors [ajoutez](#) 1 (un *nombre de sauts de ligne requis*) au début et à la fin de *items*. [\[AFFICHAGE CSS\]](#)

*Les flotteurs et les éléments en position absolue entrent dans cette catégorie.*

10. Articles de retour .

*Notez que les nœuds descendants de la plupart des éléments remplacés (par exemple, `textarea`, `input`, et `video`— mais pas `button`) ne sont pas rendus par CSS, à proprement parler, et n'ont donc pas [de cases CSS](#) pour les besoins de cet algorithme.*

Cet algorithme est susceptible d'être généralisé pour travailler sur [des pages](#). Ensuite, nous pouvons l'utiliser comme base pour [Selection](#)le stringifier de et peut-être l'exposer directement sur [les pages](#). Voir [bogue Bugzilla 10583](#).

---

Les `innerText`étapes du setter sont :

1. Soit *fragment* le [fragment de texte rendu](#) pour la valeur donnée étant donné [le document de nœud](#) de [this](#).
2. [Remplacez tout](#) par *fragment* dans [ce fichier](#).

Les `outerText`étapes du setter sont :

1. Si le parent de [this](#) est null, lancez un `"NoModificationAllowedError"` [DOMException](#).
2. Soit *ensuite* le [prochain frère](#) de [ce](#) frère.
3. Soit *précédent* le [frère précédent](#) de [this](#).

4. Soit *fragment* le fragment de texte rendu pour la valeur donnée étant donné le document de nœud de this .
5. Si *fragment* n'a pas d'enfants , alors ajoutez un nouveau Textnœud dont les données sont la chaîne vide et le nœud document est ce document de nœud à *fragmenter* .
6. Remplacez this par *fragment* dans le parent de this .
7. Si *next* n'est pas nul et que le frère précédent de *next* est un nœud, alors fusionner avec le nœud de texte suivant étant donné le frère précédent de *next* .Text
8. Si *précédent* est un Textnœud, alors fusionner avec le nœud de texte suivant étant donné *précédent* .

Le **fragment de texte rendu** pour une *entrée* de chaîne donnée à un Document document est le résultat de l'exécution des étapes suivantes :

1. Soit *fragment* un new DocumentFragment dont le nœud document est *document* .
2. Soit *position* une variable de position pour *input* , pointant initialement au début de *input* .
3. Soit *text* la chaîne vide.
4. Tant que *la position* n'est pas au-delà de la fin de *l'entrée* :
  1. Collectez une séquence de points de code qui ne sont pas U+000A LF ou U+000D CR à partir de *la position* donnée en *entrée* et définissez *le texte* sur le résultat.
  2. Si *text* n'est pas la chaîne vide, ajoutez un nouveau Textnœud dont les données sont *text* et le nœud document est *document* à *fragment* .
  3. Tant que *la position* n'est pas au-delà de la fin de *l'entrée* et que le point de code à *la position* est soit U+000A LF soit U+000D CR :
    1. Si le point de code à *la position* est U+000D CR et que le point de code suivant est U+000A LF, alors avancez *la position* jusqu'au point de code suivant dans *l'entrée* .
    2. Avancez *la position* jusqu'au point de code suivant dans *input* .
    3. Ajoutez le résultat de la création d'un élément donné *document* , br et l' espace de noms HTML à *fragment* .
5. *Fragment* de retour .

Pour **fusionner avec le nœud de texte suivant** étant donné un *nœud*[Text](#) nœud :

1. Soit *next* le [frère suivant](#) du *nœud* .
2. Si *next* n'est pas un [Text](#)nœud, retournez.
3. [Remplacez data](#) par *node* , *node* 's [data](#) 's [length](#) , 0 et *next* 's [data](#) .
4. Si le parent de *next* n'est pas nul, [supprimez next](#) .

*La vérification du parent est nécessaire car l'étape précédente peut avoir déclenché des événements de mutation.*

### 3.2.8 Exigences relatives à l'algorithme bidirectionnel

#### 3.2.8.1 Critères de conformité de création pour les caractères de formatage d'algorithme bidirectionnel

[Le contenu textuel](#) des [éléments HTML](#) avec [Text](#) des nœuds dans leur [contenu](#) et le texte des attributs des [éléments HTML](#) qui autorisent le texte de forme libre peuvent contenir des caractères dans les plages U+202A à U+202E et U+2066 à U+2069 (le bidirectionnel- caractères de formatage de l'algorithme). [\[BIDI\]](#)

*Les auteurs sont encouragés à utiliser l' [dir](#) attribut, l' [bdo](#) élément et l' [bdi](#) élément, plutôt que de conserver manuellement les caractères de mise en forme de l'algorithme bidirectionnel. Les caractères de mise en forme de l'algorithme bidirectionnel interagissent mal avec CSS.*

#### 3.2.8.2 Critères de conformité de l'agent utilisateur

Les agents utilisateurs doivent implémenter l'algorithme bidirectionnel Unicode pour déterminer le bon ordre des caractères lors du rendu de documents et de parties de documents. [\[BIDI\]](#)

Le mappage de HTML à l'algorithme bidirectionnel Unicode doit être effectué de l'une des trois manières. Soit l'agent utilisateur doit implémenter CSS, incluant notamment les propriétés CSS ['unicode-bidi'](#) , ['direction'](#) , et ['content'](#) , et doit avoir, dans sa feuille de style d'agent utilisateur, les règles utilisant ces propriétés données dans [le rendu](#) de cette spécification section, ou, alternativement, l'agent utilisateur doit agir comme s'il implémentait uniquement les propriétés susmentionnées et avait une feuille de style d'agent utilisateur qui incluait toutes les règles susmentionnées, mais sans laisser les feuilles de style spécifiées dans les documents les remplacer, ou, alternativement, l'utilisateur L'agent doit implémenter un autre langage de style avec une sémantique équivalente. [\[CSSG\]](#)

Les éléments et attributs suivants ont des exigences définies par la section [de rendu](#) qui, en raison des exigences de cette section, sont des exigences pour tous les agents utilisateurs (pas seulement ceux qui [prennent en charge le rendu par défaut suggéré](#)) :

- [dir](#)attribut
- [bdi](#)élément
- [bdo](#)élément
- [br](#)élément
- [pre](#)élément
- [textarea](#)élément
- [wbr](#)élément

### 3.2.9 Exigences liées à ARIA et aux API d'accessibilité de la plateforme

Les exigences de l'agent utilisateur pour l'implémentation de la sémantique de l'API d'accessibilité sur [les éléments HTML](#) sont définies dans *HTML Accessibility API Mappings*. En plus des règles ici, pour un élément [personnalisé element](#), la sémantique du rôle ARIA par défaut est déterminée comme suit : [\[HTMLAAM\]](#)

1. Soit *map* la [carte sémantique d'accessibilité native](#) de l'élément.
2. Si la carte [ " role" ] [existe](#), renvoyez-la.
3. Ne renvoie aucun rôle.

De même, pour un élément [personnalisé element](#), l'état ARIA par défaut et la sémantique de propriété, pour un état ou une propriété nommée *stateOrProperty*, sont déterminés comme suit :

1. Soit *map* la [carte sémantique d'accessibilité native](#) de l'élément.
2. Si la carte [ *stateOrProperty* ] [existe](#), renvoyez-la.
3. Renvoie la valeur par défaut pour *stateOrProperty*.

*La « sémantique par défaut » à laquelle il est fait référence ici est parfois également appelée sémantique « native », « implicite » ou « langue hôte » dans ARIA . [\[ARIA\]](#) Une implication de ces définitions est que la sémantique par défaut peut changer avec le temps. Cela permet aux éléments personnalisés d'avoir la même expressivité que les éléments intégrés ; par exemple, comparez à la façon dont la sémantique de rôle ARIA par défaut d'un [a](#)élément change lorsque l' [href](#)attribut est ajouté ou supprimé.*

Pour un exemple de ceci en action, voir [la section des éléments personnalisés](#).

---

Les exigences du vérificateur de conformité pour vérifier l'utilisation d'ARIA [role](#) et [aria-\\*](#) des attributs sur [les éléments HTML](#) sont définies dans *ARIA en HTML* . [\[ARIAHTML\]](#)

## 4 Les éléments du HTML

### 4.1 L'élément document

#### 4.1.1 L' [html](#) élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

[En tant qu'élément de document](#) du document .

Partout où un fragment de sous-document est autorisé dans un document composé.

##### Modèle de contenu :

Un [head](#) élément suivi d'un [body](#) élément.

##### Omission de balise dans text/html :

[La balise de début](#) d'un [html](#) élément peut être omise si la première chose à l'intérieur de l' élément n'est pas un [commentaire](#) .[html](#)

[La balise de fin](#) d'un [html](#) élément peut être omise si l' élément n'est pas immédiatement suivi d'un [commentaire](#) .[html](#)

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

##### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLHtmlElement : HTMLElement {
```



```
[HTMLConstructor] constructor();
```

```
// also has obsolete members
```

```
};
```

L' html élément représente la racine d'un document HTML.

Les auteurs sont encouragés à spécifier un lang attribut sur l'élément racine html, indiquant la langue du document. Cela aide les outils de synthèse vocale à déterminer les prononciations à utiliser, les outils de traduction à déterminer les règles à utiliser, etc.

L' html élément dans l'exemple suivant déclare que la langue du document est l'anglais.

```
<!DOCTYPE html>
<html lang="en">
<head>
<title>Swapping Songs</title>
</head>
<body>
<h1>Swapping Songs</h1>
<p>Tonight I swapped some of the songs I wrote with some friends,
who
gave me some of the songs they wrote. I love sharing my music.</p>
</body>
</html>
```

## 4.2 Métadonnées du document

### 4.2.1 L' head élément



Catégories :

Aucun.

Contextes dans lesquels cet élément peut être utilisé :

En tant que premier élément dans un html élément.

### Modèle de contenu :

Si le document est [un iframe srcdoc](#)[document](#) ou si les informations sur le titre sont disponibles à partir d'un protocole de niveau supérieur : zéro ou plusieurs éléments de [contenu de métadonnées](#) , dont pas plus d'un est un [title](#)élément et pas plus d'un est un [base](#)élément.

Sinon : un ou plusieurs éléments de [contenu de métadonnées](#) , dont exactement un est un [title](#)élément et pas plus d'un est un [base](#)élément.

### Omission de balise dans text/html :

La [balise de début](#)[head](#) d'un élément peut être omise si l'élément est vide ou si la première chose à l'intérieur de l' élément est un élément.[head](#)

La [balise de fin](#)[head](#) d'un élément peut être omise si l' élément n'est pas immédiatement suivi d' [un espace ASCII](#) ou d'un [commentaire](#) .[head](#)

### Attributs de contenu :

[Attributs globaux](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLHeadElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
};
```

L' [head](#)élément [représente](#) une collection de métadonnées pour le [Document](#).

La collection de métadonnées dans un [head](#)élément peut être grande ou petite. Voici un exemple très court :

```
<!doctype html>
<html lang=en>
  <head>
    <title>A document with a short head</title>
  </head>
  <body>
    ...
```

Voici un exemple d'un plus long:

```
<!DOCTYPE HTML>
```

```

<HTML LANG="EN">
  <HEAD>
    <META CHARSET="UTF-8">
    <BASE HREF="https://www.example.com/">
    <TITLE>An application with a long head</TITLE>
    <LINK REL="STYLESHEET" HREF="default.css">
    <LINK REL="STYLESHEET ALTERNATE" HREF="big.css" TITLE="Big Text">
    <SCRIPT SRC="support.js"></SCRIPT>
    <META NAME="APPLICATION-NAME" CONTENT="Long headed application">
  </HEAD>
  <BODY>
    ...

```

L' title élément est un enfant requis dans la plupart des situations, mais lorsqu'un protocole de niveau supérieur fournit des informations sur le titre, par exemple, dans la ligne d'objet d'un e-mail lorsque HTML est utilisé comme format de création d'e-mail, l'élément peut être omis title.

#### 4.2.2 L' title élément



##### Catégories :

Contenu des métadonnées .

##### Contextes dans lesquels cet élément peut être utilisé :

Dans un head élément ne contenant aucun autre title élément.

##### Modèle de contenu :

Texte qui n'est pas un espace entre les éléments .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

Attributs globaux

##### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

##### Interface DOM :

[Exposed=Window]

```

interface HTMLTitleElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute DOMString text;

};

```

L' title élément représente le titre ou le nom du document. Les auteurs doivent utiliser des titres qui identifient leurs documents même lorsqu'ils sont utilisés hors contexte, par exemple dans l'historique ou les signets d'un utilisateur, ou dans les résultats de recherche. Le titre du document est souvent différent de son premier titre, puisque le premier titre n'a pas à être autonome lorsqu'il est sorti de son contexte.

Il ne doit pas y avoir plus d'un title élément par document.

*S'il est raisonnable que l'élément Document n'ait pas de titre, l' title élément n'est probablement pas requis. Voir le head modèle de contenu de l'élément pour une description du moment où l'élément est requis.*

**title.text** [ = value ]

Renvoie le contenu du texte enfant de l'élément.

Peut être défini pour remplacer les enfants de l'élément par la valeur donnée.

Le **text** getter de l'attribut doit renvoyer le contenu textuel enfant title de cet élément .

Le **text** setter de l'attribut doit chaîne remplacer all par la valeur donnée dans cet title élément.

Voici quelques exemples de titres appropriés, en contraste avec les titres de niveau supérieur qui pourraient être utilisés sur ces mêmes pages.

```

<title>Introduction to The Mating Rituals of Bees</title>
...
<h1>Introduction</h1>
<p>This companion guide to the highly successful
<cite>Introduction to Medieval Bee-Keeping</cite> book is...

```

La page suivante peut faire partie du même site. Notez comment le titre décrit le sujet sans ambiguïté, tandis que le premier titre suppose que le lecteur connaît le contexte et ne se demandera donc pas si les danses sont de la salsa ou de la valse :

```

<title>Dances used during bee mating rituals</title>
...

```

```
<h1>The Dances</h1>
```

La chaîne à utiliser comme titre du document est donnée par l' [document.title](#) attribut IDL.

Les agents utilisateurs doivent utiliser le titre du document lorsqu'ils font référence au document dans leur interface utilisateur. Lorsque le contenu d'un [title](#) élément est utilisé de cette manière, [la directionnalité](#) de cet [title](#) élément doit être utilisée pour définir la directionnalité du titre du document dans l'interface utilisateur.

#### 4.2.3 L' [base](#) élément



##### Catégories :

[Contenu des métadonnées](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Dans un [head](#) élément ne contenant aucun autre [base](#) élément.

##### Modèle de contenu :

[Rien](#) .

##### Omission de balise dans text/html :

Pas [de balise de fin](#) .

##### Attributs de contenu :

[Attributs globaux](#)

[href](#) — [URL de base du document](#)

[target](#) — [Navigable](#) par défaut pour [la navigation par lien hypertexte](#) et [la](#) [soumission de formulaire](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

##### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLBaseElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
[CEReactions] attribute USVString href;  
[CEReactions] attribute DOMString target;  
};
```

L' baseélément permet aux auteurs de spécifier l' URL de base du document aux fins d' analyse des URL , et le nom de l' élément navigable par défaut aux fins de suivi des hyperliens . L'élément ne représente aucun contenu au-delà de ces informations.

Il ne doit pas y avoir plus d'un baseélément par document.

Un baseélément doit avoir soit un href attribut, soit un targetattribut, soit les deux.

L' hrefattribut content, s'il est spécifié, doit contenir une URL valide potentiellement entourée d'espaces .

Un baseélément, s'il a un hrefattribut, doit venir avant tous les autres éléments de l'arborescence qui ont des attributs définis comme prenant des URL , à l'exception de l' htmlélément (son manifestattribut n'est pas affecté par base les éléments).

*S'il y a plusieurs baseéléments avec hrefdes attributs, tous sauf le premier sont ignorés.*

L' targetattribut, s'il est spécifié, doit contenir un nom de cible navigable valide ou un mot clé , qui spécifie quel élément navigable doit être utilisé par défaut lorsque des liens hypertexte et des formulaires dans la Documentcause navigation .

Un baseélément, s'il a un target attribut, doit précéder tout élément de l'arborescence qui représente des hyperliens .

*S'il y a plusieurs baseéléments avec targetdes attributs, tous sauf le premier sont ignorés.*

Pour **obtenir la cible d'un élément** , étant donné un élément , ou elementa , areaexécutez formces étapes :

1. Si l'élément a un targetattribut, renvoie la valeur de cet attribut.
2. Si le document de nœud de l' élément contient un élément avec un attribut, renvoie la valeur de l' attribut du premier élément de ce type.basetargettargetbase
3. Renvoie la chaîne vide.

---

Un baseélément qui est le premier baseélément avec un hrefattribut de contenu dans une arborescence de documents a une **URL de base gelée** . L' URL de base gelée doit être immédiatement définie pour un élément chaque fois que l'une des situations suivantes se produit :

- L' baseélément devient le premier baseélément dans l'arborescence avec un hrefattribut de contenu dans son fichier Document.
- L' baseélément est le premier baseélément dans l'arborescence avec un hrefattribut de contenu dans son Document, et son hrefattribut de contenu est modifié.

Pour **définir l'URL de base gelée** d'un *élément* element :

1. Soit *document* le nœud document de l' *élément* .
2. Soit *urlRecord* le résultat de l'analyse de la valeur de l'attribut contenu de l' *élément*href avec l'URL de base de secours du *document* et l'encodage des caractères du *document* . (Ainsi, l' élément n'est pas affecté par lui-même.)base
3. Définissez l' URL de base gelée de l' *élément* sur l' URL de base de secours du *document* , si *urlRecord* échoue ou est en cours d'exécution. La base est-elle autorisée pour le document ? sur l' enregistrement d'URL résultant et le *document* renvoie " " , et sinon sur *urlRecord* .Blocked

L' hrefattribut IDL, lors de l'obtention, doit renvoyer le résultat de l'exécution de l'algorithme suivant :

1. Soit *document* le nœud document de l' *élément* .
2. Soit *url* la valeur de l' href attribut de cet élément, s'il en a un, et la chaîne vide sinon.
3. Soit *urlRecord* le résultat de l'analyse de l'URL avec l'URL de base de secours du *document* et l'encodage des caractères du *document* . (Ainsi, l' élément n'est pas affecté par d'autres éléments ou par lui-même.)basebase
4. Si *urlRecord* est un échec, renvoie *url* .
5. Renvoie la sérialisation de *urlRecord* .

L' hrefattribut IDL, lors de la définition, doit définir l' hrefattribut de contenu sur la nouvelle valeur donnée.

L' targetattribut IDL doit refléter l'attribut de contenu du même nom.

Dans cet exemple, un [base](#)élément est utilisé pour définir l' [URL de base du document](#) :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>This is an example for the <base>
    element</title>
    <base href="https://www.example.com/news/index.html">
  </head>
  <body>
    <p>Visit the <a href="archives.html">archives</a>.</p>
  </body>
</html>
```

Le lien dans l'exemple ci-dessus serait un lien vers  
" <https://www.example.com/news/archives.html>".

#### 4.2.4 L' [link](#)élément



##### Catégories :

[Contenu des métadonnées](#) .

Si l'élément est [autorisé dans le corps](#) : [contenu du flux](#) .

Si l'élément est [autorisé dans le corps](#) : [phrasing content](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu des métadonnées](#) est attendu.

Dans un [noscript](#)élément qui est un enfant d'un [head](#)élément.

Si l'élément est [autorisé dans le corps](#) : où [le contenu de la phrase](#) est attendu.

##### Modèle de contenu :

[Rien](#) .

##### Omission de balise dans text/html :

Pas [de balise de fin](#) .

##### Attributs de contenu :

[Attributs globaux](#)

[href](#)— Adresse du [lien hypertexte](#)

[crossorigin](#)— Comment l'élément gère les requêtes crossorigin



rel— Relation entre le document contenant l' [hyperlien](#) et la ressource de destination

media— Supports applicables

integrity— Métadonnées d'intégrité utilisées dans les contrôles *d'intégrité des sous-ressources* [\[SRI\]](#)

hreflang— Langue de la ressource liée

type— Indice pour le type de la ressource référencée

referrerpolicy— [Politique de référence](#) pour [les récupérations](#) initiées par l'élément

sizes— Tailles des icônes (pour rel=" icon")

imagesrcset— Images à utiliser dans différentes situations, par exemple, écrans haute résolution, petits moniteurs, etc. (pour rel=" preload")

imagesizes— Tailles d'image pour différentes mises en page (pour rel=" preload")

as— [Destination potentielle](#) pour une demande de préchargement (pour rel=" preload" et rel=" modulepreload")

blocking— Si l'élément est [potentiellement bloquant le rendu](#)

color— Couleur à utiliser lors de la personnalisation de l'icône d'un site (pour rel=" mask-icon")

disabled— Si le lien est désactivé

De plus, l' title attribut [a une sémantique spéciale](#) sur cet élément : Titre du lien ; [Nom du jeu de feuilles de style CSS](#) .

### **Considérations d'accessibilité :**

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### **Interface DOM :**

[Exposed=Window]

```
interface HTMLLinkElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute USVString href;
```

```
  [CEReactions] attribute DOMString? crossOrigin;
```

```
  [CEReactions] attribute DOMString rel;
```

```
  [CEReactions] attribute DOMString as;
```

```
  [SameObject, PutForwards=value] readonly attribute
```

```
    DOMTokenList relList;
```

```

[CEReactions] attribute DOMString media;

[CEReactions] attribute DOMString integrity;

[CEReactions] attribute DOMString hreflang;

[CEReactions] attribute DOMString type;

[SameObject, PutForwards=value] readonly attribute
DOMTokenList sizes;

[CEReactions] attribute USVString imageSrcset;

[CEReactions] attribute DOMString imageSizes;

[CEReactions] attribute DOMString referrerPolicy;

[SameObject, PutForwards=value] readonly attribute
DOMTokenList blocking;

[CEReactions] attribute boolean disabled;

// also has obsolete members

};

HTMLLinkElement includes LinkStyle;

```

L' [link](#) élément permet aux auteurs de lier leur document à d'autres ressources.

L'adresse du ou des liens est donnée par l' [href](#) attribut. Si l' [href](#) attribut est présent, sa valeur doit être une [URL non vide valide potentiellement entourée d'espaces](#) . L'un ou les deux attributs [href](#) ou [imagesrcset](#) doivent être présents.

Si les attributs [href](#) et [imagesrcset](#) sont absents, l'élément ne définit pas de lien.

Les types de lien indiqués (les relations) sont donnés par la valeur de l' [rel](#) attribut, qui, s'il est présent, doit avoir une valeur qui est un [ensemble non ordonné de jetons uniques séparés par des espaces](#) . Les [mots clés autorisés et leurs significations](#) sont définis dans une section ultérieure. Si l' [rel](#) attribut est absent, n'a pas de mots-clés, ou si aucun des mots-clés utilisés n'est autorisé selon les définitions de cette spécification, alors l'élément ne crée aucun lien.

rel Les jetons pris en charge par sont les mots clés définis dans les types de liens HTML qui sont autorisés sur link les éléments, ont un impact sur le modèle de traitement et sont pris en charge par l'agent utilisateur. Les jetons pris en charge sont alternate, dns-prefetch, icon, manifest, modulepreload, next, pingback, preconnect, prefetch, preload, prerender, search, et stylesheet. rel Les jetons pris en charge par doivent uniquement inclure les jetons de cette liste pour lesquels l'agent utilisateur implémente le modèle de traitement.

*Théoriquement, un agent utilisateur pourrait prendre en charge le modèle de traitement du canonical mot-clé - s'il s'agissait d'un moteur de recherche exécutant JavaScript. Mais en pratique c'est assez peu probable. Ainsi, dans la plupart des cas, canonical ne doit pas être inclus dans rel les jetons pris en charge par .*

Un link élément doit avoir soit un rel attribut, soit un itemprop attribut, mais pas les deux.

Si un link élément a un itemprop attribut, ou a un rel attribut qui ne contient que des mots-clés qui sont body-ok , alors l'élément est dit **autorisé dans le body** . Cela signifie que l'élément peut être utilisé là où un contenu de phrasé est attendu.

*Si l' rel attribut est utilisé, l'élément ne peut être utilisé qu'occasionnellement dans le body de la page. Lorsqu'il est utilisé avec l' itemprop attribut, l'élément peut être utilisé à la fois dans l' head élément et dans le body de la page, sous réserve des contraintes du modèle de microdonnées.*

---

Deux catégories de liens peuvent être créées à l'aide de l' link élément : les liens vers des ressources externes et les hyperliens . La section des types de liens définit si un type de lien particulier est une ressource externe ou un lien hypertexte. Un link élément peut créer plusieurs liens (dont certains peuvent être des liens vers des ressources externes et d'autres des hyperliens ) ; Le nombre exact de liens créés et leur nombre dépendent des mots-clés indiqués dans l' rel attribut. Les agents utilisateurs doivent traiter les liens lien par lien, et non élément par élément.

*Chaque lien créé pour un link élément est traité séparément. Par exemple, s'il y a deux link éléments avec rel="stylesheet", ils comptent chacun comme une ressource externe distincte, et chacun est affecté par ses propres attributs indépendamment. De même, si un seul link élément a un rel attribut avec la valeur next stylesheet, il crée à la fois un lien hypertexte (pour le next mot clé) et un lien de ressource externe (pour le stylesheet mot clé), et ils sont affectés différemment par d'autres attributs (tels que media ou title).*

Par exemple, l' `link` élément suivant crée deux [liens hypertexte](#) (vers la même page) :

```
<link rel="author license" href="/about">
```

Les deux liens créés par cet élément sont l'un dont la sémantique est que la page cible contient des informations sur l'auteur de la page actuelle, et l'autre dont la sémantique est que la page cible contient des informations concernant la licence sous laquelle la page actuelle est fournie.

[Les hyperliens](#) créés avec l' `link` élément et son `rel` attribut s'appliquent à l'ensemble du document. Cela contraste avec l' `rel` attribut des éléments `a` et `area` , qui indique le type d'un lien dont le contexte est donné par l'emplacement du lien dans le document.

Contrairement à ceux créés par les éléments `a` et `area`, [les hyperliens](#) créés par `link` les éléments ne sont pas affichés comme faisant partie du document par défaut, dans les agents utilisateurs qui [prennent en charge le rendu par défaut suggéré](#) . Et même s'ils sont affichés de force à l'aide de CSS, ils n'ont aucun [comportement d'activation](#) . Au lieu de cela, ils fournissent principalement des informations sémantiques qui pourraient être utilisées par la page ou par d'autres logiciels qui consomment le contenu de la page. De plus, l'agent utilisateur peut [fournir sa propre interface utilisateur pour suivre ces hyperliens](#) .

Le comportement exact des [liens vers des ressources externes](#) dépend de la relation exacte, telle que définie pour le [type de lien](#) pertinent .

---

L' `crossorigin` attribut est un [attribut de paramètres CORS](#) . Il est destiné à être utilisé avec [des liens vers des ressources externes](#) .

L' `media` attribut indique à quel média la ressource s'applique. La valeur doit être une [liste de requêtes média valide](#) .



L' `integrity` attribut représente les [métadonnées d'intégrité](#) des demandes dont cet élément est responsable. La valeur est du texte. L'attribut ne doit être spécifié que sur `link` les éléments dont un `rel` attribut contient le mot-clé `stylesheet`, `preload` ou `modulepreload`. [\[ISR\]](#)

L' `hreflang` attribut sur l' `link` élément a la même sémantique que l' `hreflang` attribut sur l' `a` élément .

L' **type**attribut donne le [type MIME](#) de la ressource liée. C'est purement consultatif. La valeur doit être une [chaîne de type MIME valide](#) .

Pour [les liens vers des ressources externes](#) , l' **type**attribut est utilisé comme indice pour les agents utilisateurs afin qu'ils puissent éviter de récupérer des ressources qu'ils ne prennent pas en charge.

L' **referrerpolicy**attribut est un [attribut de stratégie de référence](#) . Il est destiné à être utilisé avec [des liens de ressources externes](#) , où il permet de définir la [politique de référence](#) utilisée lors [de la récupération et du traitement de la ressource liée](#) . [\[POLITIQUE DE RÉFÉRENCE\]](#) .

L' **title**attribut donne le titre du lien. À une exception près, il est purement consultatif. La valeur est du texte. L'exception concerne les liens de feuille de style qui se trouvent [dans une arborescence de documents](#) , pour lesquels l' **title**attribut définit [les ensembles de feuilles de style CSS](#) .

*L' **title**attribut sur [link](#) les éléments diffère de l' **title**attribut global de la plupart des autres éléments en ce qu'un lien sans titre n'hérite pas du titre de l'élément parent : il n'a simplement pas de titre.*

---

L' **imagesrcset** attribut peut être présent et est un [attribut srcset](#) .

Les attributs **imagesrcset** et **href** (si [les descripteurs de largeur](#) ne sont pas utilisés) contribuent ensemble les [sources d'image](#) au [jeu de sources](#) .

Si l' **imagesrcset** attribut est présent et a des [chaînes candidates d'image](#) utilisant un [descripteur de largeur](#) , l' **imagesizes** attribut doit également être présent et est un [attribut de tailles](#) . L' **imagesizes** attribut contribue à la [taille de la source](#) dans l' [ensemble source](#) .

Les attributs **imagesrcset** et **imagesizes** ne doivent être spécifiés que sur [link](#) les éléments qui ont à la fois un **rel** attribut qui spécifie le **preload** mot-clé, ainsi qu'un **as** attribut dans l' `image` état " " .

Ces attributs permettent de précharger la ressource appropriée qui sera ensuite utilisée par un **img** élément ayant les valeurs correspondantes pour ses attributs **srcset** et **:sizes**

```
<link rel="preload" as="image"
      imagesrcset="wolf_400px.jpg 400w, wolf_800px.jpg 800w,
wolf_1600px.jpg 1600w"
      imagesizes="50vw">
```

```
<!-- ... later, or perhaps inserted dynamically ... -->

```

Notez comment nous omettons l' [href](#)attribut, car il ne serait pertinent que pour les navigateurs qui ne prennent pas en charge [imagesrcset](#), et dans ces cas, cela entraînerait probablement le préchargement de l'image incorrecte.

L' [imagesrcset](#)attribut peut être combiné avec l' [media](#)attribut pour précharger la ressource appropriée sélectionnée parmi [picture](#)les sources d'un élément, pour [la direction artistique](#) :

```
<link rel="preload" as="image"
      imagesrcset="dog-cropped-1x.jpg, dog-cropped-2x.jpg 2x"
      media="(max-width: 800px)">
<link rel="preload" as="image"
      imagesrcset="dog-wide-1x.jpg, dog-wide-2x.jpg 2x"
      media="(min-width: 801px)">

<!-- ... later, or perhaps inserted dynamically ... -->
<picture>
  <source srcset="dog-cropped-1x.jpg, dog-cropped-2x.jpg 2x"
          media="(max-width: 800px)">
  
</picture>
```

---

L' [sizes](#)attribut donne les tailles des icônes pour les médias visuels. Sa valeur, si elle est présente, n'est que consultative. Les agents utilisateurs peuvent utiliser la valeur pour décider quelle(s) icône(s) utiliser si plusieurs icônes sont disponibles. S'il est spécifié, l'attribut doit avoir une valeur qui est un [ensemble non ordonné de jetons uniques séparés par des espaces](#) qui ne sont [pas sensibles à la casse ASCII](#) . Chaque valeur doit être soit une correspondance [ASCII non sensible à la casse](#) pour la chaîne " [any](#) ", soit une valeur composée de deux [entiers non négatifs valides](#) qui ne sont pas précédés d'un caractère U+0030 DIGIT ZERO (0) et qui sont séparés par un seul caractère U+0078 LETTRE MINUSCULE LATINE X ou U+0058 LETTRE MAJUSCULE LATINE X. L'attribut ne doit être spécifié que sur [link](#) ayant un [rel](#)attribut qui spécifie le [icon](#)mot-clé ou le `apple-touch-icon`mot-clé.

*Le `apple-touch-icon` mot-clé est une [extension enregistrée de l'ensemble prédéfini de types de liens](#) , mais les agents utilisateurs ne sont en aucun cas tenus de le prendre en charge.*

---

L' `as` attribut spécifie la [destination potentielle](#) d'une demande de préchargement pour la ressource donnée par l' `href` attribut. C'est un [attribut énuméré](#) . Chaque [destination potentielle](#) est un mot-clé pour cet attribut, mappé à un état du même nom. L'attribut doit être spécifié sur `link` les éléments qui ont un `rel` attribut qui contient le `preload` mot-clé. Il peut être spécifié sur `link` des éléments qui ont un `rel` attribut qui contient le `modulepreload` mot-clé ; dans de tels cas, il doit avoir une valeur qui est une [destination de type script](#) . Pour les autres `link` éléments, il ne doit pas être spécifié.

Le modèle de traitement pour la façon dont l' `as` attribut est utilisé est donné dans [l'extraction et le traitement d'un type de lien individuel de l'](#) algorithme de ressource liée.

*L'attribut n'a pas de [valeur manquante par défaut](#) ou de [valeur non valide par défaut](#) , ce qui signifie que les valeurs non valides ou manquantes pour l'attribut correspondent à aucun état. Ceci est pris en compte dans le modèle de traitement. Pour `preload` les liens, les deux conditions sont une erreur ; pour `modulepreload` les liens, une valeur manquante sera traitée comme "script".*

---

L' `blocking` attribut est un [attribut bloquant](#) . Il est utilisé par le type de lien `stylesheet` et ne doit être spécifié que sur les éléments de lien qui ont un `rel` attribut contenant ce mot-clé.

---

L' `color` attribut est utilisé avec le `mask-icon` type de lien. L'attribut ne doit être spécifié que sur `link` les éléments qui ont un `rel` attribut qui contient le `mask-icon` mot-clé. La valeur doit être une chaîne qui correspond à la production CSS `<color>` , définissant une couleur suggérée que les agents utilisateurs peuvent utiliser pour personnaliser l'affichage de l'icône que l'utilisateur voit lorsqu'il épingle votre site.

Cette spécification n'a aucune exigence d'agent utilisateur pour l' colorattribut. Le `mask-icon` mot-clé est une extension enregistrée de l'ensemble prédéfini de types de liens , mais les agents utilisateurs ne sont en aucun cas tenus de le prendre en charge.

---

link les éléments ont un booléen **explicitement activé** associé . C'est d'abord faux.

L' **disabled** attribut est un attribut booléen utilisé avec le stylesheet type de lien. L'attribut ne doit être spécifié que sur link les éléments qui ont un rel attribut qui contient le stylesheet mot-clé.

Chaque fois que l' disabled attribut est supprimé, définissez l' attribut explicitement activé link de l'élément sur true.

La suppression disabled dynamique de l'attribut, par exemple à l'aide de `document.querySelector("link").removeAttribute("disabled")` , récupère et applique la feuille de style :

```
<link disabled rel="alternate stylesheet" href="css/pooh">
```

---

✓ MDN

Les attributs IDL **href**, **hreflang**, **integrity**, **media**, **rel**, **sizes**, et chacun doivent refléter type les attributs de contenu respectifs du même nom. **blockingdisabled**

Il n'y a pas d'attribut IDL reflétant pour l' color attribut, mais cela pourrait être ajouté ultérieurement.

✓ MDN

L' **as** attribut IDL doit refléter l' as attribut content, limité aux seules valeurs connues .

L' **crossOrigin** attribut IDL doit refléter l' crossorigin attribut content, limité aux seules valeurs connues .

✓ MDN



L' **referrerPolicy**attribut IDL doit [refléter](#) l' **referrerpolicy**attribut content, [limité aux seules valeurs connues](#) .

L' **imageSrcset**attribut IDL doit [refléter](#) l' **imagesrcset**attribut content.

L' **imageSizes**attribut IDL doit [refléter](#) l' **imagesizes**attribut content.



L' **relList** attribut IDL doit [refléter](#) l' **rel**attribut content.

*L' **relList**attribut peut être utilisé pour la détection de fonctionnalités, en appelant sa **supports()** méthode pour vérifier quels [types de liens](#) sont pris en charge.*

#### 4.2.4.1 Traitement de l' **media**attribut

Si le lien est un [hyperlien](#) , l' **media** attribut est purement consultatif et décrit pour quel média le document en question a été conçu.

Toutefois, si le lien est un [lien vers une ressource externe](#) , l' **media**attribut est prescriptif. L'agent utilisateur doit appliquer la ressource externe lorsque la **media**valeur de l'attribut [correspond à l'environnement](#) et que les autres conditions pertinentes s'appliquent, et ne doit pas l'appliquer autrement.

La valeur par défaut, si l' **media**attribut est omis, est " `all`", ce qui signifie que par défaut les liens s'appliquent à tous les médias.

*La ressource externe peut avoir d'autres restrictions définies dans cette limite d'applicabilité. Par exemple, une feuille de style CSS peut avoir des **@media** blocs. Cette spécification ne prévaut pas sur ces autres restrictions ou exigences.*

#### 4.2.4.2 Traitement de l' **type**attribut

Si l' **type**attribut est présent, alors l'agent utilisateur doit supposer que la ressource est du type donné (même si ce n'est pas une [chaîne de type MIME valide](#) , par exemple la chaîne vide). Si l'attribut est omis, mais que le type [de lien de ressource externe](#) a un type par défaut défini, alors l'agent utilisateur doit supposer que la ressource est de ce type. Si l'UA ne prend pas en charge le [type MIME](#) donné pour la relation de liaison donnée, alors l'UA ne devrait pas [récupérer et traiter la ressource liée](#) ; si l'UA prend en charge le [type MIME](#) donné pour la relation de lien donnée, alors l'UA doit [récupérer et traiter la ressource liée](#) au moment approprié comme

spécifié pour le type particulier de [lien de ressource externe](#) . Si l'attribut est omis et que le type [de lien de ressource externe](#) n'a pas de type par défaut défini, mais que l'agent utilisateur récupère [et traite la ressource liée](#) si le type est connu et pris en charge, alors l'agent utilisateur doit [récupérer et traiter la ressource liée](#) sous l'hypothèse qu'il sera pris en charge.

Les agents utilisateurs ne doivent pas considérer l' [type](#) attribut comme faisant autorité — lors de la récupération de la ressource, les agents utilisateurs ne doivent pas utiliser l' [type](#) attribut pour déterminer son type réel. Seul le type réel (tel que défini dans le paragraphe suivant) est utilisé pour déterminer s'il faut *appliquer* la ressource, et non le type supposé susmentionné.

Si le type [de lien de ressource externe](#) définit des règles pour le traitement des [métadonnées Content-Type](#) de la ressource , ces règles s'appliquent. Sinon, si la ressource est censée être une image, les agents utilisateurs peuvent appliquer les [règles de reniflage d'image](#) , le *type officiel* étant le type déterminé à partir des [métadonnées Content-Type](#) de la ressource , et utiliser le [type calculé résultant de la ressource](#) comme s'il était le type réel. Sinon, si aucune de ces conditions ne s'applique ou si l'agent utilisateur choisit de ne pas appliquer les règles de reniflage d'image, alors l'agent utilisateur doit utiliser les [métadonnées Content-Type de la ressource](#) pour déterminer le type de ressource. S'il n'y a pas de métadonnées de type, mais que le type [de lien de ressource externe](#) a un type par défaut défini, alors l'agent utilisateur doit supposer que la ressource est de ce type.

*Le [stylesheet](#) type de lien définit les règles de traitement des [métadonnées Content-Type](#) de la ressource .*

Une fois que l'agent utilisateur a établi le type de la ressource, l'agent utilisateur doit appliquer la ressource si elle est d'un type pris en charge et si les autres conditions pertinentes s'appliquent, et doit ignorer la ressource dans le cas contraire.

Si un document contient des liens de feuille de style étiquetés comme suit :

```
<link rel="stylesheet" href="A" type="text/plain">
<link rel="stylesheet" href="B" type="text/css">
<link rel="stylesheet" href="C">
```

... alors un UA conforme qui ne prend en charge que les feuilles de style CSS récupère les fichiers B et C et ignore le fichier A (car ce [text/plain](#) n'est pas le [type MIME](#) pour les feuilles de style CSS).

Pour les fichiers B et C, il vérifierait alors les types réels renvoyés par le serveur. Pour ceux qui sont envoyés en tant que [text/css](#), il appliquerait les styles, mais pour ceux étiquetés en tant que [text/plain](#), ou tout autre type, ce ne serait pas le cas.

Si l'un des deux fichiers était renvoyé sans métadonnée [Content-Type](#) , ou avec un type syntaxiquement incorrect comme `Content-Type: "null"`, alors le type par

défaut pour [stylesheet](#) les liens entreraient en jeu. Étant donné que ce type par défaut est [text/css](#), la feuille de style *serait* néanmoins appliquée.

#### 4.2.4.3 Récupérer et traiter une ressource à partir d'un [link](#)élément

Tous [les liens de ressources externes](#) ont une **récupération et traitement** l' algorithme de ressource liée, qui prend un [link](#)élément *el* . Ils ont également **des étapes de configuration de récupération de ressources liées** qui prennent un [link](#)élément *el* et une requête [request](#) . Les types de liens individuels peuvent fournir leur propre [extraction et traitement](#) l' algorithme de ressource liée, mais sauf indication contraire, ils utilisent l' extraction par défaut et traitent l' algorithme [de ressource liée](#) . De même, les types de liens individuels peuvent fournir leurs propres [étapes de configuration de la récupération des ressources liées](#) , mais sauf mention explicite, ces étapes renvoient simplement true.

La **récupération et le traitement par défaut de la ressource liée** , étant donné un [link](#)élément *el* , sont les suivants :

1. Soit *options* le résultat de [la création d'options de lien](#) à partir de *el* .
2. Soit *request* le résultat de [la création d'une requête de lien](#) avec *options* .
3. Si *la demande* est nulle, alors retour.
4. Définir [l'indicateur synchrone](#) de *la requête* .
5. Exécutez les [étapes de configuration de la récupération des ressources liées](#) , en indiquant *el* et *request* . Si le résultat est faux, retournez.
6. Définissez [le type d](#) ' initiateur de *la requête* sur " " si l' attribut de *el* contient le mot - clé ; " " sinon. [cssrelstylesheetlink](#)
7. [Extraire](#) *la demande* avec [processResponseConsumeBody](#) défini sur les étapes suivantes en fonction de *la réponse* [réponse](#) et null, échec ou une [séquence d'octets](#) *bodyBytes* :
  1. Que *le succès* soit vrai.
  2. Si l'une des conditions suivantes est remplie :
    - *bodyBytes* est null ou échec ; ou
    - [le statut](#) de *la réponse* n'est pas un [statut ok](#) ,puis définissez *success* sur false.

Notez que les erreurs spécifiques au contenu, par exemple les erreurs d'analyse CSS ou les erreurs de décodage PNG, n'affectent pas le succès .

3. Sinon, attendez que les [sous-ressources critiques](#) de la [ressource de lien](#) finissent de se charger.

La spécification qui définit les [sous-ressources critiques](#) d'un type de lien (par exemple CSS) est censée décrire comment ces sous-ressources sont récupérées et traitées. Cependant, comme cela n'est pas encore explicite, cette spécification décrit l'attente que [les sous-ressources critiques](#) d'une [ressource de lien](#) soient extraites et traitées, en espérant que cela se fera correctement.

4. [Traiter la ressource liée](#) donnée *el* , *success* , *response* et *bodyBytes* .

Pour **créer une demande de lien** en fonction des options [de traitement d'un lien](#) :

1. [Assert](#) : [le href](#) des *options* n'est pas la chaîne vide.
2. Si [la destination](#) des *options* n'est pas une [destination](#) , alors renvoie null.
3. [Analyser une URL](#) en fonction du [href](#) des *options* , par rapport à l'URL de [base](#) des *options* . Si cela échoue, retournez null. Sinon, laissez *url* être l' [enregistrement d'URL résultant](#) .
4. Soit *request* le résultat de [la création d'une requête CORS potentielle](#) donnée *url* , [destination](#) des *options* et [crossorigin](#) des *options* .
5. Définissez [le conteneur de politique](#) de *la requête* sur [le conteneur de politique](#) des *options* .
6. Définissez [les métadonnées d'intégrité](#) de *la requête* sur [l'intégrité](#) des *options* .
7. Définissez [les métadonnées nonce cryptographiques](#) de *la requête* sur [les métadonnées nonce cryptographiques](#) des *options* .
8. Définissez [la politique de référence](#) de *la demande* sur [la politique de référence](#) des *options* .
9. Définissez le [client de](#) *la requête* sur l' [environnement](#) des *options* .
10. *Demande* de retour .

Les agents utilisateurs peuvent choisir de n'essayer de [récupérer et de traiter](#) ces ressources que lorsqu'elles sont nécessaires, au lieu de récupérer de manière proactive toutes les [ressources externes](#) qui ne sont pas appliquées.

Semblable à l' algorithme [d'extraction et de traitement de la ressource liée](#) , tous [les liens de ressources externes](#) ont un **processus l'algorithme de ressource liée** qui prend un [link](#)élément *el* , boolean *success* , une réponse [response](#) et une [séquence d'octets](#) *bodyBytes* . Les types de liens individuels peuvent fournir leur propre [processus à l'algorithme de ressource liée](#) , mais sauf mention explicite, cet algorithme ne fait rien.

Sauf indication contraire pour un [rel](#) mot-clé donné, l'élément doit [retarder l'événement de chargement](#) du [document de nœud](#) de l'élément jusqu'à ce que toutes les tentatives de [récupération et de traitement de la ressource liée](#) et de ses [sous-ressources critiques](#) soient terminées. (Les ressources que l'agent utilisateur n'a pas encore tenté de récupérer et de traiter, par exemple, parce qu'il attend que la ressource soit nécessaire, ne [retardez pas l'événement load](#) .)

#### 4.2.4.4 Traitement [Link](#) des en-têtes ``

Tous les types de liens qui peuvent être [des liens de ressources externes](#) définissent un **processus d'un** algorithme d'en-tête de lien, qui prend des [options de traitement de lien](#) . Cet algorithme définit si et comment ils réagissent à leur apparition dans un [Link](#) en-tête de réponse HTTP ``.

*Pour la plupart des types de liens, cet algorithme ne fait rien. Le [tableau récapitulatif](#) est une bonne référence pour savoir rapidement si un type de lien a défini les étapes [de traitement d'un en-tête de lien](#) .*

Une **option de traitement de lien** est une [structure](#) . Il comporte les [éléments](#) suivants :

***href*** (par défaut la chaîne vide)

***destination*** (par défaut la chaîne vide)

***initiateur*** (par défaut " [link](#) ")

***intégrité*** (par défaut la chaîne vide)

***type*** (par défaut la chaîne vide)

***métadonnées nonce cryptographiques*** (par défaut, la chaîne vide)

Un string

***crossorigin*** ( [pas de CORS](#) par défaut )

Un état [d'attribut de paramètres CORS](#)

***politique de référence*** (par défaut, la chaîne vide)

Une [politique de référencement](#)

***jeu de sources*** (null par défaut)

Null ou un [ensemble source](#)

***URL de base***

Une [URL](#)

**origine**

Une [origine](#)

**environnement**

Un [environnement](#)

**conteneur de stratégie**

Un [conteneur de politique](#)

**document (null par défaut)**

Nul ou un [Document](#)

**sur le document prêt (null par défaut)**

Null ou un algorithme acceptant un [Document](#)

*Une [option de traitement de lien](#) a une [URL de base](#) et un [href](#) plutôt qu'une URL analysée, car l'URL peut être le résultat de l' [ensemble source](#) des options .*

Pour **créer des options de lien à partir d'un élément** donné un [link](#)élément *el* :

1. Soit *document* le [nœud document](#) de *el* .
2. Soit *options* une nouvelle [option de traitement de lien](#) avec

**[destination](#)**

le résultat de [la traduction](#) de l'état de l' attribut *el*/[as](#)

**[origine croisée](#)**

l'état de l' attribut de contenu de *el*/[crossorigin](#)

**[politique de référence](#)**

l'état de l' attribut de contenu de *el*/[referrerpolicy](#)

**[jeu de sources](#)**

[ensemble source](#) d' *el*

**[URL de base](#)**

[URL](#) du *document*

**[origine](#)**

[origine](#) du *document*

**[environnement](#)**

[objet de paramètres pertinent](#) du *document*

**[conteneur de stratégie](#)**

[conteneur de stratégie](#) du *document*

**[document](#)**

*document*

**[métadonnées nonce cryptographiques](#)**

La valeur actuelle de l'emplacement interne [\[\[CryptographicNonce\]\]](#) d' *el*

3. Si *el* a un [href](#)attribut, alors définissez [le href](#) de *options* sur la valeur de l'attribut de *el* .[href](#)

4. Si *el* a un `integrity` attribut, alors définissez [l'intégrité](#) de *options* sur la valeur de l'attribut content de *el* `.integrity`
5. Si *el* a un `type` attribut, alors définissez [le type](#) de *options* sur la valeur de l'attribut de *el* `.type`
6. [Assert](#) : [le href](#) des *options* n'est pas la chaîne vide, ou [le jeu source](#) des *options* n'est pas nul.

Un [link](#) élément sans un `href` ni un `imagesrcset` ne représente pas un lien.

7. *Options* de retour .

Pour **extraire des liens à partir d'en-têtes** donnés un [en-tête liste](#) *les en-têtes* :

1. Laissez *les liens* être une nouvelle [liste](#) .
2. Soit *rawLinkHeaders* le résultat de [l'obtention, du décodage et de la séparation de](#) `Link` de [la liste d'en-têtes](#) de la *réponse* .
3. [Pour chaque](#) *linkHeader* de *rawLinkHeaders* :
  1. Soit *linkObject* le résultat de [l'analyse](#) de *linkHeader* . [\[LIEN WEB\]](#)
  2. Si *linkObject*["`target_uri`"] n'existe pas , alors [continuez](#) .
  3. [Ajouter](#) *linkObject* aux *liens* .
4. *Liens* de retour .

Pour **traiter les en-têtes de lien** avec un [Document](#) *doc* , une *réponse* [response](#) , et une *phase* " " ou " " : `pre-media`

1. Soit *les liens* le résultat de [l'extraction des liens](#) de [la liste d'en-tête](#) de la *réponse* .
2. [Pour chaque](#) *linkObject* dans *les liens* :
  1. Soit *rel* un *linkObject* ["`relation_type`"].
  2. Soit *les attributs* être *linkObject* ["`target_attributes`"].
  3. Supposons que *ExpectPhase* soit "`media`" si "`srcset`", "`imagesrcset`" ou "`media`" [existent](#) dans *les attributs* ; sinon "`pre-media`".
  4. Si la *phase attendue* n'est pas la *phase* , [continuez](#) .
  5. Si *attribs* ["`media`"] [existe](#) et *attribs* ["`media`"] ne [correspond pas à l'environnement](#) , alors [continuez](#) .



6. Soit *options* une nouvelle [option de traitement de lien](#) avec

[href](#)  
*lienObjet*[" [target\\_uri](#)"]  
[URL de base](#)  
[URL](#) du *document*  
[origine](#)  
[origine](#) du *doc*  
[environnement](#)  
[objet de paramètres pertinent](#) de *doc*  
[conteneur de stratégie](#)  
[conteneur de stratégie](#) de *doc*  
[document](#)  
*doc*

7. [Appliquez les options de lien des attributs d'en-tête analysés](#) aux *options* données aux *attributs* .
8. Si *attribs* [" [imagesrcset](#)"] [existe](#) et *attribs* [" [imagesizes](#)"] [existe](#) , alors définissez le [jeu source](#) des *options* sur le résultat de la [création d'un jeu source](#) donné *linkObject* [" "], *attribs* [" "] et *attribs* [" "].[target\\_uriimagesrcsetimagesizes](#)
9. Exécutez le [processus d'](#) étapes d'en-tête de lien pour *les options rel* données .

Pour **appliquer des options de lien à partir d'attributs d'en-tête analysés** à des *options* [d'options de traitement de lien](#) avec des *attributs* :

1. Si *les attributs* [" [as](#)"] [existent](#) , définissez [la destination](#) des *options* sur le résultat de [la traduction](#) des *attributs* [" "]. [as](#)
2. Si *attribs* [" [crossorigin](#)"] [existe](#) et correspond à une correspondance [ASCII insensible à la casse](#) pour l'un des [mots-clés d'attribut de paramètres CORS](#) , définissez [l'origine croisée](#) des *options* sur l' état [d'attribut de paramètres CORS](#) correspondant à ce mot-clé.
3. Si *attribs* [" [integrity](#)"] [existe](#) , alors définissez [l'intégrité](#) des *options* sur *attribs* [" "].[integrity](#)
4. Si *attribs* [" [referrerpolicy](#)"] [existe](#) et correspond à une correspondance [ASCII insensible à la casse](#) pour une [politique de référent](#) , alors définissez [la politique de référent](#) des *options* sur cette [politique de référent](#) .
5. Si *attribs* [" [nonce](#)"] [existe](#) , alors définissez [le nonce](#) des *options* sur *attribs* [" "].[nonce](#)
6. Si *attribs* [" [type](#)"] [existe](#) , alors définissez [le type](#) des *options* sur *attribs* [" "].[type](#)



#### 4.2.4.5 Premiers indices



**Les premiers conseils** permettent aux agents utilisateurs d'effectuer certaines opérations, telles que le chargement spéculatif de ressources susceptibles d'être utilisées par le document, avant que la demande de navigation ne soit entièrement traitée par le serveur et qu'un code de réponse ne soit servi. Les serveurs peuvent indiquer des indices précoces en servant une [réponse](#) avec un code d'état 103 avant de servir la [réponse](#) finale . [\[RFC8297\]](#)

Par exemple, étant donné la séquence de réponses suivante :

```
103 Premier indice
Lien : </image.png> ; rel = précharge ; comme = image
200 D'accord
Type de contenu : texte/html

<!DOCTYPE html>
...

```

l'image commencera à se charger avant que le contenu HTML n'arrive.

*Seule la première réponse d'indication anticipée servie pendant la navigation est gérée, et elle est ignorée si elle est suivie d'une redirection d'origine croisée.*

En plus des [Linken-têtes](#) ``, il est possible que la réponse 103 contienne un en-tête [Content Security Policy](#) , qui est appliqué lors du traitement de l'indice précoce.

Par exemple, étant donné la séquence de réponses suivante :

```
103 Premier indice
Content-Security-Policy: style-src: self;
Lien : </style.css> ; rel = précharge ; comme =style
103 Premier indice
Lien : </image.png> ; rel = précharge ; comme = image
302 Redirection
Emplacement : /alternate.html
200 D'accord
Content-Security-Policy : style-src : aucun ;
Lien : </font.ttf> ; rel = précharge ; comme =police
```

La police et le style seraient chargés et l'image serait supprimée, car seule la première réponse d'indice précoce dans la chaîne de redirection finale est respectée. L' en-tête [de politique de sécurité du contenu](#) tardif vient après que la

demande de récupération du style a déjà été effectuée, mais le style ne sera pas accessible au document.

Pour **traiter les premiers en-têtes d'indication** donnés une *réponse* [response](#) et un [environnement](#) *reservedEnvironment* :

*Les en-têtes [Link](#) à indice précoce sont toujours traités avant [Link](#) les en-têtes de la [réponse](#) finale, suivis des [link](#) éléments. Cela équivaut à ajouter le contenu des premiers et derniers [Link](#) en-têtes à l'[Document](#) élément [head](#)', dans l'ordre respectif.*

1. Supposons que *earlyPolicyContainer* soit le résultat de [la création d'un conteneur de stratégie à partir d'une réponse de récupération](#) donnée *response* et *reservedEnvironment* .

*Cela permet à la [réponse](#) d'indice précoce d'inclure une [politique de sécurité du contenu](#) qui serait [appliquée](#) lors de la récupération de la [demande](#) d'indice précoce .*

2. Soit *les liens* le résultat de [l'extraction des liens](#) de [la liste d'en-tête](#) de la *réponse* .
3. Soit *earlyHints* une [liste](#) vide .
4. [Pour chaque](#) *linkObject* dans *les liens* :

*Au moment où nous recevons l'en-tête du lien d'indication précoce, nous commençons à [récupérer](#) *earlyRequest* . S'il revient avant que le [Document](#) soit créé, nous définissons *earlyResponse* sur la [réponse](#) de cette [récupération](#) et une fois que le [Document](#) est créé, nous le validons (en le rendant disponible dans la [carte des ressources préchargées](#) comme s'il s'agissait d'un [link](#) élément). Si le [Document](#) est créé en premier, la [réponse](#) est validée dès qu'elle devient disponible.*

1. Soit *rel* un *linkObject* ["*relation\_type*"].
2. Soit *options* une nouvelle [option de traitement de lien](#) avec

[href](#)  
*lienObjet* ["*target\_uri*"]  
[initiateur](#)  
"*early-hint*"  
[URL de base](#)  
[URL](#) de la *réponse*  
[origine](#)  
[origine](#) de l' [URL](#) de la *réponse*  
[environnement](#)  
*environnement réservé*  
[conteneur de stratégie](#)  
*earlyPolicyContainer*  
*earlyPolicyContainer*

3. Soit les attributs être `linkObject["target_attributes"]`.

Seuls les attributs `src`, `srcset`, et `integrity` sont gérés dans le cadre du traitement des as premiers conseils `crossorigin`. Les autres, notamment `alt`, `loading`, `onload`, et `onerror`, ne sont applicables qu'une fois a créé. `integrity` `type` `blocking` `imagesrcset` `imagesizes` `mediaDo` `cument`

4. Appliquez les options de lien des attributs d'en-tête analysés aux *options* données aux *attributs* .
5. Exécutez le processus d' étapes d'en-tête de lien pour les *options rel* données .
6. Ajoutez des *options* à *earlyHints* .
5. Renvoyez les sous-étapes suivantes données Document *doc* : pour chaque *option* dans *earlyHints* :
  1. Si *options* 's on document ready est null, alors définissez le document d' *options* sur *doc* s.
  2. Sinon, appelez *options* sur document ready avec *doc* .

#### 4.2.4.6 Fournir aux utilisateurs un moyen de suivre les hyperliens créés à l'aide de l' `link` élément

Les agents utilisateurs interactifs peuvent fournir aux utilisateurs un moyen de [suivre les hyperliens](#) créés à l'aide de l' `link` élément, quelque part dans leur interface utilisateur. L'interface exacte n'est pas définie par cette spécification, mais elle pourrait inclure les informations suivantes (obtenues à partir des attributs de l'élément, à nouveau comme défini ci-dessous), sous une forme ou une autre (éventuellement simplifiée), pour chaque lien hypertexte créé avec chaque élément `link` dans le document :

- La relation entre ce document et la ressource (donnée par l' rel attribut).
- Le titre de la ressource (donné par l' title attribut).
- L'adresse de la ressource (donnée par l' href attribut).
- La langue de la ressource (donnée par l' hreflang attribut).
- Le support optimal pour la ressource (donné par l' media attribut).

Les agents utilisateurs peuvent également inclure d'autres informations, telles que le type de la ressource (tel qu'il est donné par l' `type` attribut).

## 4.2.5 L' **meta** élément



### Catégories :

Contenu des métadonnées .

Si l' itemprop attribut est présent : contenu du flux .

Si l' itemprop attribut est présent : phrasing content .

### Contextes dans lesquels cet élément peut être utilisé :

Si l' charset attribut est présent, ou si l' http-equiv attribut de l'élément est à l' état Encoding declaration : dans un head élément.

Si l' http-equiv attribut est présent mais pas à l' état Encoding declaration : dans un head élément.

Si l' http-equiv attribut est présent mais pas dans l' état Encoding declaration : dans un noscript élément qui est un enfant d'un head élément.

Si l' name attribut est présent : où le contenu des métadonnées est attendu.

Si l' itemprop attribut est présent : où le contenu des métadonnées est attendu.

Si l' itemprop attribut est présent : où le contenu de la phrase est attendu.

### Modèle de contenu :

Rien .

### Omission de balise dans text/html :

Pas de balise de fin .

### Attributs de contenu :

Attributs globaux

name— Nom des métadonnées

http-equiv— Directive Pragma

content— Valeur de l'élément

charset— Déclaration de codage de caractères

media— Supports applicables

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLMetaElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```

[CEReactions] attribute DOMString name;

[CEReactions] attribute DOMString httpEquiv;

[CEReactions] attribute DOMString content;

[CEReactions] attribute DOMString media;

// also has obsolete members

};

```

L' metaélément représente divers types de métadonnées qui ne peuvent pas être exprimées à l'aide des éléments title, base, link, style et script.

L' metaélément peut représenter des métadonnées au niveau du document avec l' name attribut, des directives pragma avec l' attribut et la déclaration d'encodage de caractères http-equiv du fichier lorsqu'un document HTML est sérialisé sous forme de chaîne (par exemple pour la transmission sur le réseau ou pour le stockage sur disque) avec l' attribut charset.

Exactement l'un des attributs name, http-equiv, charset et itemprop doit être spécifié.

Si name, http-equiv ou itemprop est spécifié, l' content attribut doit également être spécifié. Sinon, il doit être omis.

L' charset attribut spécifie le codage de caractères utilisé par le document. Il s'agit d'une déclaration d'encodage de caractères. Si l'attribut est présent, sa valeur doit être une correspondance ASCII non sensible à la casse pour la chaîne " utf-8".

*L' charset attribut sur l' metaélément n'a aucun effet dans les documents XML, mais est autorisé dans les documents XML afin de faciliter la migration vers et depuis XML.*

Il ne doit pas y avoir plus d'un metaélément avec un charset attribut par document.

L' content attribut donne la valeur des métadonnées du document ou de la directive pragma lorsque l'élément est utilisé à ces fins. Les valeurs autorisées dépendent du contexte exact, comme décrit dans les sections suivantes de cette spécification.

Si un metaélément a un name attribut, il définit les métadonnées du document. Les métadonnées du document sont exprimées en termes de paires nom-valeur, l' name attribut sur l' metaélément donnant le nom et l' content attribut sur le même élément donnant la valeur. Le nom spécifie quel aspect des métadonnées est

défini ; les noms valides et la signification de leurs valeurs sont décrits dans les sections suivantes. Si un [meta](#)élément n'a pas [content](#)d'attribut, la partie valeur de la paire nom-valeur des métadonnées est la chaîne vide.

L' [media](#)attribut indique à quel média les métadonnées s'appliquent. La valeur doit être une [liste de requêtes média valide](#) . Sauf si [name](#)is [theme-color](#), l' [media](#) attribut n'a aucun effet sur le modèle de traitement et ne doit pas être utilisé par les auteurs.

Les attributs [name](#), [content](#)et [media](#)IDL doivent [refléter](#) les attributs de contenu respectifs du même nom. L'attribut IDL [httpEquiv](#)doit [refléter](#) l'attribut content [http-equiv](#).

#### 4.2.5.1 Noms de métadonnées standards



Cette spécification définit quelques noms pour l' [name](#) attribut de l' [meta](#)élément.

Les noms sont insensibles à la casse et doivent être comparés d'une manière [ASCII insensible à la casse](#) .

##### [application-name](#)

La valeur doit être une courte chaîne de forme libre indiquant le nom de l'application Web représentée par la page. Si la page n'est pas une application Web, le [application-name](#)nom des métadonnées ne doit pas être utilisé. Des traductions du nom de l'application Web peuvent être données, en utilisant l' [lang](#)attribut pour spécifier la langue de chaque nom.

Il ne doit pas y avoir plus d'un [meta](#)élément avec une [langue](#) donnée et où la [name](#)valeur de l'attribut est une correspondance [ASCII non sensible à la casse](#) pour [application-name](#)chaque document.

Les agents utilisateurs peuvent utiliser le nom de l'application dans l'interface utilisateur de préférence à celui de la page [title](#), car le titre peut inclure des messages d'état et autres concernant l'état de la page à un moment donné au lieu d'être simplement le nom de l'application.

Pour trouver le nom de l'application à utiliser dans une liste ordonnée de langues (par exemple, l'anglais britannique, l'anglais américain et l'anglais), les agents utilisateurs doivent exécuter les étapes suivantes :

1. Soit *langues* la liste des langues.

2. Soit *langue par défaut* la [langue](#) de l' [élément de document](#)[Document](#) de , le cas échéant, et si cette langue n'est pas inconnue.
3. S'il existe une *langue par défaut* , et si ce n'est pas la même langue que l' une des langues dans *langues* , ajoutez - la à *langues* .
4. Supposons que *la langue gagnante* soit la première langue parmi les *langues* pour lesquelles il existe un [meta](#)élément dans [Document](#) où la [name](#)valeur de l'attribut est une correspondance [ASCII insensible à la casse](#)[application-name](#) pour et dont [la langue](#) est la langue en question.

Si aucune des langues n'a un tel [meta](#)élément, alors retournez ; il n'y a pas de nom d'application donné.

5. Renvoie la valeur de l' [content](#)attribut du premier [meta](#)élément dans [Document](#) l' [arborescence](#) où la [name](#)valeur de l'attribut est une correspondance [ASCII non sensible à la casse](#)[application-name](#) pour et dont [la langue](#) est *la langue gagnante* .

*Cet algorithme serait utilisé par un navigateur lorsqu'il a besoin d'un nom pour la page, par exemple, pour étiqueter un signet. Les langues qu'il fournirait à l'algorithme seraient les langues préférées de l'utilisateur.*

#### **author**

La valeur doit être une chaîne libre indiquant le nom de l'un des auteurs de la page.

#### **description**

La valeur doit être une chaîne de forme libre qui décrit la page. La valeur doit être appropriée pour une utilisation dans un répertoire de pages, par exemple dans un moteur de recherche. Il ne doit pas y avoir plus d'un [meta](#)élément où la [name](#)valeur d'attribut est une correspondance [ASCII non sensible à la casse](#) pour [description](#) chaque document.

#### **generator**

La valeur doit être une chaîne de forme libre qui identifie l'un des progiciels utilisés pour générer le document. Cette valeur ne doit pas être utilisée sur les pages dont le balisage n'est pas généré par un logiciel, par exemple les pages dont le balisage a été écrit par un utilisateur dans un éditeur de texte.

Voici ce qu'un outil appelé "Frontweaver" pourrait inclure dans sa sortie, dans l' [head](#)élément de la page, pour s'identifier comme l'outil utilisé pour générer la page :

```
<meta name=generator content="Frontweaver 8.2">
```

#### **keywords**

La valeur doit être un [ensemble de jetons séparés par des virgules](#) , chacun étant un mot clé pertinent pour la page.

Cette page sur les polices de caractères sur les autoroutes britanniques utilise un [meta](#)élément pour spécifier certains mots-clés que les utilisateurs pourraient utiliser pour rechercher la page :

```
<!DOCTYPE HTML>
<html lang="en-GB">
  <head>
    <title>Typefaces on UK motorways</title>
    <meta name="keywords" content="british,type
face,font,fonts,highway,highways">
  </head>
  <body>
    ...
```

*De nombreux moteurs de recherche ne prennent pas en compte ces mots-clés, car cette fonctionnalité a toujours été utilisée de manière peu fiable, voire trompeuse, comme moyen de spammer les résultats des moteurs de recherche d'une manière qui n'est pas utile pour les utilisateurs.*

Pour obtenir la liste des mots clés que l'auteur a spécifiés comme applicables à la page, l'agent utilisateur doit exécuter les étapes suivantes :

1. Laissez *les mots-clés* être une liste vide.
2. Pour chaque [meta](#)élément avec un [name](#) attribut et un [content](#)attribut et où la [name](#)valeur de l'attribut est une correspondance [ASCII non sensible à la casse](#) pour [keywords](#):
  1. [Fractionnez la valeur de l' \[content\]\(#\)attribut de l'élément par des virgules](#) .
  2. Ajoutez les jetons résultants, le cas échéant, aux *mots-clés* .
3. Supprimez tous les doublons des *mots clés* .
4. Retourner *les mots-clés* . Il s'agit de la liste des mots clés que l'auteur a spécifiés comme applicables à la page.

Les agents utilisateurs ne devraient pas utiliser ces informations lorsqu'il n'y a pas suffisamment de confiance dans la fiabilité de la valeur.

Par exemple, il serait raisonnable qu'un système de gestion de contenu utilise les informations sur les mots-clés des pages du système pour remplir l'index d'un moteur de recherche spécifique à un site, mais un agrégateur de contenu à grande échelle qui utilise ces informations trouverait probablement que certains les utilisateurs essaieraient de déjouer son mécanisme de classement en utilisant des mots-clés inappropriés.



## referrer

La valeur doit être une [stratégie de référence](#) , qui définit la [stratégie de référence](#) par défaut pour le [Document](#). [\[POLITIQUE DE RÉFÉRENCE\]](#)

Si un [élément](#)[meta](#) element est [inséré dans le document](#) , ou a ses attributs ou modifiés, les agents utilisateurs doivent exécuter l'algorithme suivant :[namecontent](#)

1. Si *l'élément* n'est pas [dans une arborescence de documents](#) , alors retournez.
2. Si *l'élément* n'a pas d' [name](#) attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour " [referrer](#) ", alors retour.
3. Si *l'élément* n'a pas d' [content](#) attribut, ou si la valeur de cet attribut est la chaîne vide, alors retournez.
4. Soit *value* la valeur de l' attribut *element*[content](#) , [convertie en ASCII minuscule](#) .
5. Si *valeur* est l'une des valeurs indiquées dans la première colonne du tableau suivant, définissez *valeur* sur la valeur indiquée dans la deuxième colonne :

Valeur héritée	Politique de parrainage
never	<a href="#">no-referrer</a>
default	la <a href="#">politique de référence par défaut</a>
always	<a href="#">unsafe-url</a>
origin-when-crossorigin	<a href="#">origin-when-cross-origin</a>

6. Si *la valeur* est une [politique de référence](#) , alors définissez [la politique de référence](#) du [conteneur de politique](#) du [document du nœud](#) de l' [élément](#) sur [politique](#) .

*Pour des raisons historiques, contrairement aux autres noms de métadonnées standard, le modèle de traitement de [referrer](#) ne répond pas aux suppressions d'éléments et n'utilise pas [l'ordre de l'arborescence](#) . Seul l'élément le plus récemment inséré ou le plus récemment modifié [meta](#) dans cet état a un effet.*

## theme-color

### MDN

La valeur doit être une chaîne qui correspond à la production CSS [<color>](#) , définissant une couleur suggérée que les agents utilisateurs doivent utiliser pour personnaliser l'affichage de la page ou de l'interface utilisateur environnante. Par exemple, un navigateur peut colorer la barre de titre de la

page avec la valeur spécifiée ou l'utiliser comme surbrillance de couleur dans une barre d'onglets ou un sélecteur de tâches.

Dans un document HTML, la [media](#) valeur d'attribut doit être unique parmi tous les [meta](#) éléments avec leur [name](#) valeur d'attribut définie sur une correspondance [ASCII insensible à la casse](#) pour [theme-color](#).

Cette norme elle-même utilise "WHATWG green" comme couleur de thème :

```
<!DOCTYPE HTML>
<title>HTML Standard</title>
<meta name="theme-color" content="#3c790a">
...
```

L' [media](#) attribut peut être utilisé pour décrire le contexte dans lequel la couleur fournie doit être utilisée.

Si nous voulions uniquement utiliser "WHATWG green" comme couleur de thème de cette norme en mode sombre, nous pourrions utiliser la `prefers-color-scheme` fonction multimédia :

```
<!DOCTYPE HTML>
<title>HTML Standard</title>
<meta name="theme-color" content="#3c790a" media="(prefers-color-scheme: dark)">
...
```

Pour obtenir la couleur du thème d'une page, les agents utilisateurs doivent exécuter les étapes suivantes :

1. Soit *les éléments candidats* la liste de tous [meta](#) les éléments qui répondent aux critères suivants, dans [l'ordre de l'arborescence](#) :
  - L'élément est [dans une arborescence de documents](#)
  - L'élément a un [name](#) attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour [theme-color](#)
  - L'élément a un [content](#) attribut
2. Pour chaque *élément des éléments candidats* :
  - Si *l'élément* a un [media](#) attribut et que la valeur de l'attribut de l' *élément* [media](#) ne [correspond pas à l'environnement](#) , alors [continuez](#) .
  - Soit *value* le résultat de [la suppression des espaces blancs ASCII de début et de fin](#) de la valeur de l'attribut de l'*élément* [content](#) .
  - Soit *color* le résultat de [l'analyse](#) de *value* .

- Si la couleur n'est pas un échec, alors retourne *color* .

3. Ne renvoie rien (la page n'a pas de couleur de thème).

Si des *meta*éléments sont [insérés dans le document](#) ou [supprimés du document](#) , ou si des éléments existants *meta*voient leurs attributs *name*, *content*, ou *media* modifiés, ou si l'environnement change de sorte que la valeur d'un attribut *meta*d'élément *media* peut maintenant ou ne plus [correspondre à l'environnement](#) , les agents utilisateurs doivent re - exécuter l'algorithme ci-dessus et appliquer le résultat à toute interface utilisateur affectée.

Lors de l'utilisation de la couleur du thème dans l'interface utilisateur, les agents utilisateurs peuvent l'ajuster de manière spécifique à l'implémentation pour la rendre plus adaptée à l'interface utilisateur en question. Par exemple, si un agent utilisateur a l'intention d'utiliser la couleur du thème comme arrière-plan et d'afficher du texte blanc dessus, il peut utiliser une variante plus sombre de la couleur du thème dans cette partie de l'interface utilisateur, pour assurer un contraste adéquat.

#### **color-scheme**

Pour aider les agents utilisateurs à restituer immédiatement l'arrière-plan de la page avec le schéma de couleurs souhaité (plutôt que d'attendre que tous les CSS de la page se chargent), une valeur '[color-scheme](#)' peut être fournie dans un *meta*élément.

La valeur doit être une chaîne qui correspond à la syntaxe de la valeur de la propriété CSS '[color-scheme](#)' . Il détermine les [schémas de couleurs pris en charge par la page](#) .

Il ne doit pas y avoir plus d'un *meta*élément avec sa *name*valeur d'attribut définie sur une correspondance [ASCII non sensible à la casse](#) pour [color-scheme](#) chaque document.

La déclaration suivante indique que la page connaît et peut gérer un jeu de couleurs avec des couleurs d'arrière-plan sombres et des couleurs de premier plan claires :

```
<meta name="color-scheme" content="dark">
```

Pour obtenir [les schémas de couleurs pris en charge par une page](#) , les agents utilisateurs doivent exécuter les étapes suivantes :

1. Soit *les éléments candidats* la liste de tous *meta*les éléments qui répondent aux critères suivants, dans [l'ordre de l'arborescence](#) :
  - L'élément est [dans une arborescence de documents](#)
  - L'élément a un *name*attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour [color-scheme](#)

- L'élément a un contentattribut

2. Pour chaque *élément* des *éléments candidats* :

- Soit *parsed* le résultat de [l'analyse d'une liste de valeurs de composants](#) compte tenu de la valeur de l'attribut *element*content .
- Si *parsed* est une valeur de propriété CSS '[color-scheme](#)' valide , alors renvoie *parsed* .

3. Renvoie nul.

Si des metaéléments sont [insérés dans le document](#) ou [supprimés du document](#) , ou metasi des éléments existants voient leurs attributs nameou content modifiés, les agents utilisateurs doivent réexécuter l'algorithme ci-dessus.

*Étant donné que ces règles vérifient les éléments successifs jusqu'à ce qu'ils trouvent une correspondance, un auteur peut fournir plusieurs de ces valeurs pour gérer le repli des agents utilisateurs hérités. Contrairement au fonctionnement du retour CSS pour les propriétés, les multiples éléments méta doivent être organisés avec les valeurs héritées après les nouvelles valeurs.*

#### 4.2.5.2 Autres noms de métadonnées

N'importe qui peut créer et utiliser ses propres **extensions pour l'ensemble prédéfini de noms de métadonnées** . Il n'y a aucune obligation d'enregistrer de telles extensions.

Cependant, un nouveau nom de métadonnées ne doit pas être créé dans les cas suivants :

- Si soit le nom est une [URL](#) , soit la valeur de son contentattribut associé est une [URL](#) ; dans ces cas, il est recommandé de l'enregistrer en tant [qu'extension de l'ensemble prédéfini de types de liens](#) (plutôt que de créer un nouveau nom de métadonnées).
- Si le nom est pour quelque chose qui devrait avoir des exigences de traitement dans les agents utilisateurs ; dans ce cas, il doit être normalisé.

De plus, avant de créer et d'utiliser un nouveau nom de métadonnées, il est recommandé de consulter la page WHATWG Wiki MetaExtensions - pour éviter de choisir un nom de métadonnées déjà utilisé, et pour éviter de dupliquer l'objectif des noms de métadonnées déjà utilisés, et pour [éviter](#) nouveaux noms standardisés en conflit avec le nom que vous avez choisi. [\[WHATWGWIKI\]](#)

N'importe qui est libre de modifier la page WHATWG Wiki MetaExtensions à tout moment pour ajouter un nom de métadonnées. De nouveaux noms de métadonnées peuvent être spécifiés avec les informations suivantes :

### **Mot-clé**

Le nom réel en cours de définition. Le nom ne doit pas prêter à confusion avec tout autre nom défini (par exemple, ne différant que par la casse).

### **Brève description**

Une brève description non normative de la signification du nom des métadonnées, y compris le format dans lequel la valeur doit être.

### **spécification**

Un lien vers une description plus détaillée de la sémantique et des exigences du nom des métadonnées. Il peut s'agir d'une autre page du wiki ou d'un lien vers une page externe.

### **Synonymes**

Une liste d'autres noms qui ont exactement les mêmes exigences de traitement. Les auteurs ne doivent pas utiliser les noms définis comme synonymes (ils sont uniquement destinés à permettre aux agents utilisateurs de prendre en charge le contenu hérité). N'importe qui peut supprimer les synonymes qui ne sont pas utilisés dans la pratique ; seuls les noms qui doivent être traités comme des synonymes pour la compatibilité avec le contenu hérité doivent être enregistrés de cette manière.

### **Statut**

L'un des éléments suivants :

#### **Proposé**

Le nom n'a pas fait l'objet d'un large examen et d'une approbation par les pairs. Quelqu'un l'a proposé et l'utilise ou l'utilisera bientôt.

#### **Ratifié**

Le nom a reçu un large examen et approbation par les pairs. Il a une spécification qui définit sans ambiguïté comment gérer les pages qui utilisent le nom, y compris lorsqu'elles l'utilisent de manière incorrecte.

#### **Discontinué**

Le nom des métadonnées a fait l'objet d'un large examen par les pairs et s'est avéré insuffisant. Les pages existantes utilisent ce nom de métadonnées, mais les nouvelles pages doivent l'éviter. Les entrées "brève description" et "spécification" donneront des détails sur ce que les auteurs devraient utiliser à la place, le cas échéant.

Si un nom de métadonnées s'avère redondant avec des valeurs existantes, il doit être supprimé et répertorié comme synonyme de la valeur existante.

Si un nom de métadonnées est ajouté dans l'état "proposé" pendant une période d'un mois ou plus sans être utilisé ou spécifié, il peut être supprimé de la page WHATWG Wiki MetaExtensions.

Si un nom de métadonnées est ajouté avec le statut "proposé" et s'avère redondant avec les valeurs existantes, il doit être supprimé et répertorié comme synonyme de la valeur existante. Si un nom de métadonnée est ajouté avec le statut « proposé » et s'avère nuisible, il doit être remplacé par le statut « discontinué ».

N'importe qui peut modifier le statut à tout moment, mais ne doit le faire que conformément aux définitions ci-dessus.

#### 4.2.5.3 Directives pragmatiques

Lorsque l' **http-equiv** attribut est spécifié sur un **meta** élément, l'élément est une directive pragma.

L' **http-equiv** attribut est un **attribut énuméré** . Le tableau suivant répertorie les mots clés définis pour cet attribut. Les états donnés dans la première cellule des lignes avec des mots-clés donnent les états auxquels ces mots-clés correspondent. Certains des mots clés ne sont pas conformes, comme indiqué dans la dernière colonne.

État	Mot-clé	Remarques
<a href="#">Langue du contenu</a>	<b>content-language</b>	Non conforme
<a href="#">Déclaration d'encodage</a>	<b>content-type</b>	
<a href="#">Style par défaut</a>	<b>default-style</b>	
<a href="#">Rafraîchir</a>	<b>refresh</b>	
<a href="#">Set-Cookie</a>	<b>set-cookie</b>	Non conforme
<a href="#">Compatible X-UA</a>	<b>x-ua-compatible</b>	
<a href="#">Politique de sécurité du contenu</a>	<b>content-security-policy</b>	

Lorsqu'un **meta** élément est [inséré dans le document](#) , si son **http-equiv** attribut est présent et représente l'un des états ci-dessus, alors l'agent utilisateur doit exécuter l'algorithme approprié pour cet état, comme décrit dans la liste suivante :

**État de la langue du contenu ( )** **http-equiv="content-language"**

*Cette fonctionnalité n'est pas conforme. Les auteurs sont encouragés à utiliser l' **lang** attribut à la place.*

Ce pragma définit la **langue par défaut de pragma-set** . Jusqu'à ce qu'un tel pragma soit traité avec succès, il n'y a pas [de langue par défaut définie par pragma](#) .

1. Si l' metaélément n'a pas content d'attribut, alors retournez.
2. Si l' content attribut de l'élément contient un caractère virgule U+002C (,) alors retour.
3. Soit *input* la valeur de l' content attribut de l'élément.
4. Laissez *position* pointer sur le premier caractère de *input* .
5. Ignorer les espaces blancs ASCII dans *la position d'entrée* donnée .
6. Collectez une séquence de points de code qui ne sont pas des espaces blancs ASCII à partir de *la position* donnée en entrée .
7. Soit *candidat* la chaîne résultant de l'étape précédente.
8. Si *candidat* est la chaîne vide, retour.
9. Définissez la langue par défaut de pragma-set sur *candidate* .

*Si la valeur se compose de plusieurs jetons séparés par des espaces, les jetons après le premier sont ignorés.*

*Ce pragma est presque, mais pas tout à fait, totalement différent de l'en-tête HTTP Content-Language du même nom. [HTTP]*

#### **Etat de la déclaration d'encodage ( ) http-equiv="content-type"**

L' état de la déclaration d'encodage n'est qu'une forme alternative de définition de l' charset attribut : il s'agit d'une déclaration d'encodage de caractères . Les exigences de l'agent utilisateur de cet état sont toutes gérées par la section d'analyse de la spécification.

Pour metales éléments avec un http-equiv attribut dans l' état de déclaration Encoding , l' content attribut doit avoir une valeur qui est une correspondance ASCII insensible à la casse pour une chaîne composée de : la chaîne littérale " `text/html;` ", éventuellement suivie d'un nombre quelconque d' espaces ASCII , suivi de la chaîne littérale " `charset=utf-8` ".

Un document ne doit pas contenir à la fois un metaélément avec un http-equiv attribut dans l' état de déclaration Encoding et un metaélément avec l' charset attribut présent.

L' état de déclaration Encoding peut être utilisé dans les documents HTML , mais les éléments avec un http-equiv attribut dans cet état ne doivent pas être utilisés dans les documents XML .

#### **État de style par défaut ( ) http-equiv="default-style"**



Ce pragma définit le nom du jeu de feuilles de style CSS par défaut .

1. Si l'metaélément n'a pas content d'attribut, ou si la valeur de cet attribut est la chaîne vide, alors retournez.
2. [Modifiez le nom du jeu de feuilles de style CSS préféré](#) , le nom étant la valeur de l' contentattribut de l'élément. [\[CSSOM\]](#)

### **État d'actualisation ( )** `http-equiv="refresh"`

Ce pragma agit comme une redirection temporisée.

Un Documentobjet a un **rafraîchissement déclaratif** associé (un booléen). C'est d'abord faux.

1. Si l'metaélément n'a pas content d'attribut, ou si la valeur de cet attribut est la chaîne vide, alors retournez.
2. Soit *input* la valeur de l' contentattribut de l'élément.
3. Exécutez les [étapes d'actualisation déclarative partagée](#) avec le nœud documentmeta de l'élément , *input* et l' élément.meta

Les **étapes d'actualisation déclarative partagée** , étant donné un Documentobjet *document* , une entrée de chaîne et éventuellement un metaélément *meta* , sont les suivantes :

1. Si [l'actualisation déclarative](#) du *document* est vraie, alors retournez.
2. Laissez *position* pointer au premier [point de code](#) de *input* .
3. [Ignorer les espaces blancs ASCII](#) dans *la position d'entrée* donnée .
4. Soit *le temps* égal à 0.
5. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en *entrée* et laissez le résultat être *timeString* .
6. Si *timeString* est la chaîne vide, alors :
  1. Si le [point de code](#) dans *l'entrée* pointée par *la position* n'est pas U+002E (.), alors retour.
7. Sinon, définissez *time* sur le résultat de l'analyse de *timeString* en utilisant les [règles d'analyse des entiers non négatifs](#) .
8. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) et des caractères U+002E FULL STOP (.) à partir de *la position* donnée en *entrée* . Ignorez tous les caractères collectés.
9. Soit *urlRecord* l' [URL](#) du *document* .
10. Si *la position* n'est pas au-delà de la fin de *l'entrée* , alors :



1. Si le [point de code](#) dans l'entrée pointée par *position* n'est pas U+003B (;), U+002C (,) ou [ASCII whitespace](#) , alors retour.
2. [Ignorer les espaces blancs ASCII](#) dans la *position d'entrée* donnée .
3. Si le [point de code](#) en entrée pointé par *position* est U+003B (;) ou U+002C (,) , alors avancez *position* jusqu'au [point de code](#) suivant .
4. [Ignorer les espaces blancs ASCII](#) dans la *position d'entrée* donnée .

11. Si la *position* n'est pas au-delà de la fin de l'entrée , alors :

1. Soit *urlString* la sous-chaîne d' entrée du [point de code](#) à la *position* jusqu'à la fin de la chaîne.
2. Si le [point de code](#) en entrée pointé par *position* est U+0055 (U) ou U+0075 (u), alors avancez *position* jusqu'au [point de code](#) suivant . Sinon, passez à l'étape intitulée *sauter les guillemets* .
3. Si le [point de code](#) dans l'entrée pointé par *position* est U+0052 (R) ou U+0072 (r), alors avancez la *position* jusqu'au [point de code](#) suivant . Sinon, passez à l'étape intitulée *parse* .
4. Si le [point de code](#) en entrée pointé par *position* est U+004C (L) ou U+006C (l), alors avancez la *position* jusqu'au [point de code](#) suivant . Sinon, passez à l'étape intitulée *parse* .
5. [Ignorer les espaces blancs ASCII](#) dans la *position d'entrée* donnée .
6. Si le [point de code](#) en entrée pointé par *position* est U+003D (=), alors avancez *position* jusqu'au [point de code](#) suivant . Sinon, passez à l'étape intitulée *parse* .
7. [Ignorer les espaces blancs ASCII](#) dans la *position d'entrée* donnée .
8. *Ignorer les guillemets* : si le [point de code](#) dans l'entrée pointée par la *position* est U+0027 (') ou U+0022 (") , alors laissez la *citation* être ce [point de code](#) et avancez la *position* jusqu'au [point de code](#) suivant . Sinon, laissez la *citation* être la chaîne vide.
9. Définissez *urlString* sur la sous-chaîne d' entrée du [point de code](#) à la *position* jusqu'à la fin de la chaîne.
10. Si *quote* n'est pas la chaîne vide et qu'il existe un [point de code](#) dans *urlString* égal à *quote* , alors tronquez *urlString* à

ce [point de code](#) , de sorte qu'il et tous les points de [code](#) suivants soient supprimés.

11. *Parse* : [Parse](#) *urlString* par rapport au *document* . Si cela échoue, revenez. Sinon, définissez *urlRecord* sur l' [enregistrement d'URL résultant](#) .

12. Définissez [l'actualisation déclarative](#) du *document* sur *true*.

13. Effectuez une ou plusieurs des étapes suivantes :

1. Après l'échéance de l'actualisation (comme défini ci-dessous), si l'utilisateur n'a pas annulé la redirection et, si *meta* est donné, [l'indicateur de sandboxing actif](#) du *document* n'a pas l' [indicateur de contexte de navigation des fonctionnalités automatiques en bac à sable](#) , alors [navigatez dans](#) le *document* ' s [nœud navigable](#) vers *urlRecord* en utilisant *document* , avec [historyHandling](#) défini sur " ". [replace](#)

Aux fins du paragraphe précédent, un rafraîchissement est dit arrivé à échéance dès que la *dernière* des deux conditions suivantes se produit :

- Au moins *le temps* secondes s'est écoulé depuis [le temps de chargement complet](#) du *document* , ajusté pour prendre en compte les préférences de l'utilisateur ou de l'agent utilisateur.
- Si *meta* est donné, au moins *le temps* secondes s'est écoulé depuis que *meta* a été [inséré dans](#) le *document* , ajusté pour prendre en compte les préférences de l'utilisateur ou de l'agent utilisateur.

*Il est important d'utiliser document ici, et non [le nœud document](#) de meta , car cela peut avoir changé entre l'ensemble initial d'étapes et l'actualisation arrivant à échéance et meta n'est pas toujours donné (dans le cas de l' en-tête HTTP ``[Refresh](#)*

2. Fournit à l'utilisateur une interface qui, lorsqu'elle est sélectionnée, [navigate dans](#) le nœud *document* [navigable](#) vers *urlRecord* à l'aide de *document* .

3. Ne fais rien.

De plus, l'agent utilisateur peut, comme pour tout, informer l'utilisateur de tous les aspects de son fonctionnement, y compris l'état de tous les temporisateurs, les destinations de toutes les redirections temporisées, etc.

Pour [meta](#) les éléments avec un [http-equiv](#) attribut à l' [état Actualisé](#) , l' [content](#) attribut doit avoir une valeur consistant soit en :

- juste un [entier non négatif valide](#) , ou
- un [entier non négatif valide](#) , suivi d'un caractère U+003B SEMICOLON (;), suivi d'un ou plusieurs [espaces ASCII](#) , suivi d'une sous-chaîne correspondant à une correspondance [ASCII insensible à la casse](#) pour la chaîne " [URL](#)", suivi d'un U+ 003D SIGNE ÉGAL (=), suivi d'une [chaîne d'URL valide](#) qui ne commence pas par un caractère littéral U+0027 APOSTROPHE (') ou U+0022 QUOTATION MARK (").

Dans le premier cas, l'entier représente un nombre de secondes avant que la page ne soit rechargée ; dans ce dernier cas, l'entier représente un nombre de secondes avant que la page ne soit remplacée par la page à l' [URL](#) donnée .

La page d'accueil d'un organe de presse peut inclure le balisage suivant dans l' [head](#)élément de la page, pour s'assurer que la page se recharge automatiquement à partir du serveur toutes les cinq minutes :

```
<meta http-equiv="Refresh" content="300">
```

Une séquence de pages peut être utilisée comme un diaporama automatisé en faisant en sorte que chaque page s'actualise à la page suivante dans la séquence, en utilisant un balisage comme celui-ci :

```
<meta http-equiv="Refresh" content="20; URL=page4.html">
```

### **Définir l'état des cookies ( )** `http-equiv="set-cookie"`

Ce pragma est non conforme et n'a aucun effet.

Les agents utilisateurs doivent ignorer ce pragma.

### **État compatible X-UA ( )** `http-equiv="x-ua-compatible"`

En pratique, ce pragma incite Internet Explorer à suivre au plus près les spécifications.

Pour [meta](#)les éléments avec un [http-equiv](#) attribut dans l' [état X-UA-Compatible](#) , l' [content](#)attribut doit avoir une valeur qui est une correspondance [ASCII non sensible à la casse](#) pour la chaîne " `IE=edge`".

Les agents utilisateurs doivent ignorer ce pragma.

### **État de la politique de sécurité du contenu ( )** `http-equiv="content-security-policy"`

Ce pragma [applique](#) une [politique de sécurité du contenu](#) sur un [Document](#). [\[CSP\]](#)

1. Si l' [meta](#)élément n'est pas un enfant d'un [head](#)élément, retournez.
2. Si l' [meta](#)élément n'a pas [content](#) d'attribut, ou si la valeur de cet attribut est la chaîne vide, alors retournez.

3. Soit *politique* le résultat de l'exécution de l'analyse de la politique de sécurité du contenu [d'un algorithme de politique de sécurité du contenu sérialisé](#) sur la valeur de l'attribut [meta](#) de l'élément [content](#), avec une source de "meta" et une disposition de "enforce".
4. Supprimez toutes les occurrences des [directives](#) [report-uri](#), [frame-ancestors](#) et de la stratégie [.sandbox](#)
5. [Appliquer la](#) *politique* politique .

Pour [meta](#) les éléments avec un [http-equiv](#) attribut dans l' [état Politique de sécurité du contenu](#) , l' [content](#) attribut doit avoir une valeur consistant en une [politique de sécurité du contenu valide](#) , mais ne doit contenir aucune [directive](#) [report-uri](#) , [frame-ancestors](#) ou . La [politique de sécurité du contenu](#) indiquée dans l' attribut sera [appliquée](#) au document actuel. [\[CSP\]](#) [sandbox](#) [content](#)

*Au moment de l'insertion de l' [meta](#) élément dans le document, il est possible que certaines ressources aient déjà été récupérées. Par exemple, les images peuvent être stockées dans la [liste des images disponibles](#) avant l'insertion dynamique d'un [meta](#) élément avec un [http-equiv](#) attribut dans l' [état de la politique de sécurité du contenu](#) . Il n'est pas garanti que les ressources qui ont déjà été récupérées soient bloquées par une [politique de sécurité du contenu](#) appliquée tardivement.*

Une page peut choisir d'atténuer le risque d'attaques de scripts intersites en empêchant l'exécution de JavaScript en ligne, ainsi qu'en bloquant tout le contenu du plug-in, à l'aide d'une politique telle que la suivante :

```
<meta http-equiv="Content-Security-Policy" content="script-src 'self'; object-src 'none'">
```

Il ne doit pas y avoir plus d'un [meta](#) élément avec un état particulier dans le document à la fois.

#### 4.2.5.4 Spécification de l'encodage des caractères du document

Une **déclaration de codage de caractères** est un mécanisme par lequel le [codage de caractères](#) utilisé pour stocker ou transmettre un document est spécifié.

La norme de codage nécessite l'utilisation du [codage de caractères UTF-8](#) et nécessite l'utilisation de l' [étiquette de codage](#) " " pour l'identifier. Ces exigences nécessitent que la [déclaration de codage de caractères](#) du document , si elle existe, spécifie une [étiquette de codage](#) utilisant une correspondance [ASCII insensible à la casse](#) pour " ". Qu'une [déclaration de codage de caractères](#) soit présente ou non, le [codage de caractères](#) réel utilisé pour coder le document doit être [UTF-8](#) . [\[CODAGE\]](#) `utf-8utf-8`

Pour appliquer les règles ci-dessus, les outils de création doivent utiliser par défaut [UTF-8](#) pour les documents nouvellement créés.

Les restrictions suivantes s'appliquent également :

- La déclaration de codage de caractères doit être sérialisée sans l'utilisation de [références de caractères](#) ou d'échappements de caractères d'aucune sorte.
- L'élément contenant la déclaration de codage de caractères doit être entièrement sérialisé dans les 1024 premiers octets du document.

De plus, en raison d'un certain nombre de restrictions sur [meta](#)les éléments, il ne peut y avoir qu'une seule [meta](#)déclaration d'encodage de caractères basée sur - par document.

Si un [document HTML](#) ne commence pas par un BOM et que son [encodage](#) n'est pas explicitement donné par [Content-Type metadata](#) , et que le document n'est pas [un iframe srcdoc document](#) , alors l'encodage doit être spécifié à l'aide d'un [meta](#)élément avec un [charset](#)attribut ou d'un [meta](#)élément avec un [http-equiv](#) attribut dans l' [état Encoding declaration](#) .

*Une déclaration de codage de caractères est requise (soit dans les [métadonnées Content-Type](#), soit explicitement dans le fichier) même lorsque tous les caractères sont dans la plage ASCII, car un codage de caractères est nécessaire pour traiter les caractères non ASCII entrés par l'utilisateur dans les formulaires, dans URL générées par des scripts, etc.*

*L'utilisation d'encodages non-UTF-8 peut avoir des résultats inattendus sur la soumission de formulaires et les encodages d'URL, qui utilisent l' [encodage de caractères du document](#) par défaut.*

Si le document est [un iframe srcdoc document](#) , le document ne doit pas avoir de [déclaration de codage de caractères](#) . (Dans ce cas, la source est déjà décodée, puisqu'elle fait partie du document qui contenait le [iframe](#).)

En XML, la déclaration XML doit être utilisée pour les informations de codage de caractères en ligne, si nécessaire.

En HTML, pour déclarer que l'encodage des caractères est [UTF-8](#) , l'auteur peut inclure le balisage suivant en haut du document (dans l' [head](#)élément) :

```
<meta charset="utf-8">
```

En XML, la déclaration XML serait utilisée à la place, tout en haut du balisage :

```
<?xml version="1.0" encoding="utf-8"?>
```

## 4.2.6 L' `style` élément



### Catégories :

Contenu des métadonnées .

### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu des métadonnées est attendu.

Dans un `noscript` élément qui est un enfant d'un `head` élément.

### Modèle de contenu :

Texte qui donne une feuille de style conforme .

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

`media`— Supports applicables

`blocking`— Si l'élément est potentiellement bloquant le rendu

De plus, l' `title` attribut a une sémantique spéciale sur cet élément : CSS style sheet set name .

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLStyleElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  attribute boolean disabled;
```

```
  [CEReactions] attribute DOMString media;
```

```
  [SameObject, PutForwards=value] readonly attribute
```

```
    DOMTokenList blocking;
```

```
  // also has obsolete members
```

```
};
```

```
HTMLStyleElement includes LinkStyle;
```

L' style élément permet aux auteurs d'intégrer des feuilles de style CSS dans leurs documents. L' style élément est l'une des nombreuses entrées du modèle de traitement de style. L'élément ne représente pas le contenu pour l'utilisateur.



Les disabled étapes du getter sont :

1. S'il n'y a pas de feuille de style CSS associée , renvoyez false.
2. Si l'indicateur de désactivation de cette feuille de style CSS associée est défini, renvoie true.
3. Renvoie faux.

Les disabled étapes du setter sont :

1. Si cela n'a pas de feuille de style CSS associée , retournez.
2. Si la valeur donnée est true , définissez l' indicateur désactivé de cette feuille de style CSS associée . Sinon, désactivez l' indicateur de désactivation de cette feuille de style CSS associée .

Il est important de noter que disabled les affectations d'attributs ne prennent effet que lorsque l' style élément a une feuille de style CSS associée :

```
const style = document.createElement('style');
style.disabled = true;
style.textContent = 'body { background-color: red; }';
document.body.append(style);
console.log(style.disabled); // false
```

L' media attribut indique à quel média les styles s'appliquent. La valeur doit être une liste de requêtes média valide . L'agent utilisateur doit appliquer les styles lorsque la media valeur de l'attribut correspond à l'environnement et que les autres conditions pertinentes s'appliquent, et ne doit pas les appliquer autrement.

*Les styles peuvent être encore plus limités dans leur portée, par exemple dans CSS avec l'utilisation de @media blocs. Cette spécification ne prévaut pas sur ces autres restrictions ou exigences.*

La valeur par défaut, si l' media attribut est omis, est " all ", ce qui signifie que les styles s'appliquent par défaut à tous les médias.



L' **blocking** attribut est un [attribut bloquant](#) .



L' **title**attribut sur [style](#) les éléments définit [les ensembles de feuilles de style CSS](#) . Si l' [style](#)élément n'a pas **title**d'attribut, alors il n'a pas de titre ; l' **title**attribut des ancêtres ne s'applique pas à l' [style](#) élément. Si l' [style](#)élément n'est pas [dans une arborescence de documents](#) , l' **title**attribut est ignoré. [\[CSSOM\]](#)

*L' **title**attribut sur [style](#) les éléments, comme l' **title**attribut sur [link](#) les éléments, diffère de l' **title**attribut global en ce qu'un [style](#)bloc sans titre n'hérite pas du titre de l'élément parent : il n'a simplement pas de titre.*

Le [contenu du texte enfant](#) d'un [style](#)élément doit être celui d'une [feuille de style conforme](#) .

Un [style](#)élément est [implicitement potentiellement bloquant le rendu](#) s'il a été créé par l'analyseur de son [nœud document](#) .

---

L'agent utilisateur doit exécuter l' algorithme [de mise à jour d' \[style\]\(#\)un bloc](#) chaque fois que l'une des conditions suivantes se produit :

- L'élément est extrait de la [pile d'éléments ouverts](#) d'un [analyseur HTML](#) ou [d'un analyseur XML](#) .
- L'élément n'est pas sur la [pile d'éléments ouverts](#) d'un [analyseur HTML](#) ou [d'un analyseur XML](#) , et il [devient connecté](#) ou [déconnecté](#) .
- [Les enfants](#) de l'élément ont modifié les étapes exécutées.

La **mise à jour d'un algorithme [style](#)de bloc** est la suivante :

1. Soit *élément* l' [style](#)élément.
2. Si l'*élément* a une [feuille de style CSS associée](#) , [supprimez la feuille de style CSS](#) en question.
3. Si l'*élément* n'est pas [connecté](#) , alors retournez.
4. Si l'attribut de l' *élément*[type](#) est présent et que sa valeur n'est ni la chaîne vide ni une correspondance [ASCII insensible à la casse](#) pour " [text/css](#) ", alors retour.



*En particulier, une type valeur avec des paramètres, tels que " text/css; charset=utf-8", entraînera le retour anticipé de cet algorithme.*

5. Si le [comportement en ligne de l'élément doit-il être bloqué par la politique de sécurité du contenu ?](#) algorithm renvoie " Blocked" lorsqu'il est exécuté sur l' styleélément, " style" et le [contenu textuel enfant](#)style de l'élément , puis renvoie. [\[CSP\]](#)
6. [Créez une feuille de style CSS](#) avec les propriétés suivantes :

[taper](#)

[text/css](#)

[nœud propriétaire](#)

élément

[médias](#)

L' [media](#)attribut de l'élément .

*Il s'agit d'une référence à l'attribut (éventuellement absent à ce moment), plutôt qu'une copie de la valeur actuelle de l'attribut. CSSOM définit ce qui se passe lorsque l'attribut est défini, modifié ou supprimé de manière dynamique.*

[titre](#)

L' [title](#)attribut de *element* , si *element* est [dans une arborescence de documents](#) , ou la chaîne vide sinon.

*Encore une fois, il s'agit d'une référence à l'attribut.*

[drapeau alternatif](#)

Désactivé.

[drapeau d'origine propre](#)

Ensemble.

[emplacement](#)

[feuille de style CSS mère](#)

[règle CSS du propriétaire](#)

nul

[drapeau désactivé](#)

Laissé à sa valeur par défaut.

[Règles CSS](#)

Laissé non initialisé.

Cela ne semble pas juste. Vraisemblablement, nous devrions utiliser le contenu textuel enfant de l'élément ? Suivi en tant que numéro 2997 .

7. Si l'élément contribue à une feuille de style de blocage de script , ajoutez élément à l' ensemble de feuilles de style de blocage de script de son document de nœud .
8. Si la valeur de l'attribut de l' élément correspond à l'environnement et que l'élément est potentiellement bloquant le rendu , alors bloquez le rendu sur l'élément .media

Une fois que les tentatives d'obtention des sous-ressources critiques de la feuille de style , le cas échéant, sont terminées, ou, si la feuille de style n'a pas de sous-ressources critiques , une fois que la feuille de style a été analysée et traitée, l'agent utilisateur doit exécuter ces étapes :

La récupération des sous-ressources critiques n'est pas bien définie ; le numéro 968 est probablement la meilleure résolution pour cela. Dans l'intervalle, toute demande de sous-ressource critique doit avoir son paramètre de blocage de rendu défini sur le fait que l' élément soit ou non en train de bloquer le rendu .style

1. Soit *element* l' .styleélément associé à la feuille de style en question.
2. Que *le succès* soit vrai.
3. Si les tentatives d'obtention de l'une des sous-ressources critiques de la feuille de style échouent pour une raison quelconque (par exemple, erreur DNS, réponse HTTP 404, connexion prématurément fermée, type de contenu non pris en charge), définissez *success* sur false.

*Notez que les erreurs spécifiques au contenu, par exemple les erreurs d'analyse CSS ou les erreurs de décodage PNG, n'affectent pas le succès .*

4. Mettez en file d'attente une tâche d'élément sur la source de tâche de mise en réseau de l'élément donné et procédez comme suit :
  1. Si *le succès* est vrai, déclenchez un événement nommé load à l'élément .
  2. Sinon, déclenchez un événement nommé error à *element* .
  3. Si l'élément contribue à une feuille de style bloquant les scripts :
    1. Assert : le jeu de feuilles de style de blocage de script du document de nœud de l'élément contient l'élément .
    2. Supprimer l'élément de l'ensemble de feuilles de style de blocage de script de son document de nœud .

#### 4. [Débloquez le rendu](#) sur l'élément .

L'élément doit [retarder l'événement de chargement](#) du [document de nœud](#) de l'élément jusqu'à ce que toutes les tentatives d'obtention des [sous-ressources critiques](#) de la feuille de style , le cas échéant, soient terminées.

*Cette spécification ne spécifie pas de système de style, mais CSS devrait être pris en charge par la plupart des navigateurs Web. [\[CSS\]](#)*

✓ MDN

Les attributs **media** et **blocking** IDL doivent chacun [refléter](#) les attributs de contenu respectifs du même nom.

L' [LinkStyle](#) interface est également implémentée par cet élément. [\[CSSOM\]](#)

Le document suivant a son emphase accentuée sous forme de texte rouge vif plutôt que de texte en italique, tout en laissant les titres d'œuvres et les mots latins dans leur italique par défaut. Il montre comment l'utilisation d'éléments appropriés permet un restylage plus facile des documents.

```
<!DOCTYPE html>
<html lang="en-US">
  <head>
    <title>My favorite book</title>
    <style>
      body { color: black; background: white; }
      em { font-style: normal; color: red; }
    </style>
  </head>
  <body>
    <p>My <em>favorite</em> book of all time has <em>got</em> to be
    <cite>A Cat's Life</cite>. It is a book by P. Rahmel that talks
    about the <i lang="la">Felis catus</i> in modern human
    society.</p>
  </body>
</html>
```

### 4.2.7 Interactions du style et du script

Si la feuille de style ne fait référence à aucune autre ressource (par exemple, c'était une feuille de style interne donnée par un [style](#) élément sans `@import` règles), alors les règles de style doivent être [immédiatement](#) mises à la disposition du script

; sinon, les règles de style ne doivent être mises à la disposition du script qu'une fois que la [boucle d'événement](#) atteint sa [mise à jour à l'étape de rendu](#) .

Un élément *e/* dans le contexte de *a d'* [Document](#) un [analyseur HTML](#) ou [d'un analyseur XML](#) **contribue à une feuille de style bloquant les scripts** si toutes les conditions suivantes sont vraies :

- *e/* a été créé par cet [Document](#) analyseur.
- *e/* est soit un [style](#) élément, soit un [link](#) élément qui était un [lien de ressource externe qui contribue au modèle de traitement de style](#) lorsque *e/* a été créé par l'analyseur.
- *media* La valeur de l'attribut *el* *correspond* [à l'environnement](#) .
- *La feuille de style de e/* a été activée lorsque l'élément a été créé par l'analyseur.
- La dernière fois que la [boucle d'événements](#) a atteint [l'étape 1](#) , [la racine](#) de *e/* était celle-là . [Document](#)
- L'agent utilisateur n'a pas encore renoncé à charger cette feuille de style particulière. Un agent utilisateur peut renoncer à charger une feuille de style à tout moment.

*Abandonner une feuille de style avant le chargement de la feuille de style, si la feuille de style finit toujours par se charger, signifie que le script peut finir par fonctionner avec des informations incorrectes. Par exemple, si une feuille de style définit la couleur d'un élément sur vert, mais qu'un script qui inspecte le style résultant est exécuté avant le chargement de la feuille, le script trouvera que l'élément est noir (ou quelle que soit la couleur par défaut), et pourrait donc faire de mauvais choix (par exemple, décider d'utiliser le noir comme couleur ailleurs sur la page, au lieu du vert). Les implémenteurs doivent équilibrer la probabilité qu'un script utilise des informations incorrectes avec l'impact sur les performances de ne rien faire en attendant qu'une requête réseau lente se termine.*

On s'attend à ce que les contreparties des règles ci-dessus s'appliquent également aux `<?xml-stylesheet?>`PI. Cependant, cela n'a pas encore fait l'objet d'une enquête approfondie.

A [Document](#) possède un **jeu de feuilles de style bloquant les scripts** , qui est un [jeu ordonné](#) , initialement vide.

Un [Document](#) *document* a une **feuille de style qui bloque les scripts** si les étapes suivantes renvoient true :

1. Si [le jeu de feuilles de style de blocage de script](#) du *document* n'est pas [vide](#) , alors renvoie true.

2. Si [le nœud navigable](#) du *document* est null, alors renvoie false.
3. Soit *containerDocument* le *document* [conteneur](#) du [noeud navigable](#) du document .
4. Si *containerDocument* n'est pas null et que [l'ensemble de feuilles de style de blocage de script](#) de *containerDocument* n'est pas [vide](#) , alors renvoie true.
5. Renvoie faux.

A [Document](#) n'a pas de feuille de style qui bloque les scripts s'il n'a pas [de feuille de style qui bloque les scripts](#) .

## 4.3 Rubriques



### 4.3.1 L' [body](#) élément



#### Catégories :

Aucun.

#### Contextes dans lesquels cet élément peut être utilisé :

En tant que deuxième élément dans un [html](#) élément.

#### Modèle de contenu :

[Contenu du flux](#) .

#### Omission de balise dans text/html :

La [balise de début](#)[body](#) d'un élément peut être omise si l'élément est vide, ou si la première chose à l'intérieur de l' élément n'est pas [un espace ASCII](#) ou un [commentaire](#) , sauf si la première chose à l'intérieur de l' élément est un élément , , , , ou . [bodybodymetanoscRIPTlinkscriptstyletemplate](#)

La [balise de fin](#)[body](#) d'un élément peut être omise si l' élément n'est pas immédiatement suivi d'un [commentaire](#) .[body](#)

#### Attributs de contenu :

[Attributs globaux](#)

[onafterprint](#)

[onbeforeprint](#)

[onbeforeunload](#)

[onhashchange](#)  
[onlanguagechange](#)  
[onmessage](#)  
[onmessageerror](#)  
[onoffline](#)  
[ononline](#)  
[onpagehide](#)  
[onpageshow](#)  
[onpopstate](#)  
[onrejectionhandled](#)  
[onstorage](#)  
[onunhandledrejection](#)  
[onunload](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### Interface DOM :

[Exposed=Window]

interface **HTMLBodyElement** : [HTMLElement](#) {

[[HTMLConstructor](#)] constructor();

// [also has obsolete members](#)

};

[HTMLBodyElement](#) includes [WindowEventHandlers](#);

L' [body](#)élément [représente](#) le contenu du document.

Dans les documents conformes, il n'y a qu'un seul [body](#)élément. L' [document.body](#)attribut IDL fournit aux scripts un accès facile à l' [body](#)élément d'un document.

*Certaines opérations DOM (par exemple, des parties du modèle [de glisser-déposer](#) ) sont définies en termes de " [l'élément de corps](#) ". Cela fait référence à un élément particulier dans le DOM, selon la définition du terme, et non à un [body](#)élément arbitraire.*

L' [body](#)élément expose en tant [qu'attributs de contenu de gestionnaire d'événements](#) un certain nombre de [gestionnaires d'événements](#) de l' [Window](#)objet. Il reflète également leurs [attributs IDL de gestionnaire d'événements](#) .

Les [gestionnaires d'événements](#) de l' [Window](#) objet nommé par l' [Window](#) ensemble de [gestionnaires d'événements d'élément de corps -reflecting](#) , exposés sur l' [body](#) élément, remplacent les [gestionnaires d'événements](#) génériques par les mêmes noms normalement pris en charge par [les éléments HTML](#) .

Ainsi, par exemple, un événement bouillonnant [error](#) distribué sur un enfant de l'[élément body](#) de [Document](#) a déclencherait d'abord les [onerror attributs de contenu du gestionnaire d'événements](#) de cet élément, puis celui de l' [html](#) élément racine, et *alors* seulement il déclencherait l' [onerror attribut de contenu du gestionnaire d'événements](#) sur le [body](#) élément. C'est parce que l'événement remonterait de la cible, vers le [body](#), vers le [html](#), vers le [Document](#), vers le [Window](#), et le [gestionnaire d'événements](#) sur le [body](#) surveille le [Window](#) pas le [body](#). Un écouteur d'événement régulier attaché à l' [body](#) utilisation `addEventListener()` , cependant, serait exécuté lorsque l'événement a traversé [body](#) et non lorsqu'il atteint l' [Window](#) objet. Cette page met à jour un indicateur pour indiquer si l'utilisateur est en ligne ou non :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>Online or offline?</title>
    <script>
      function update(online) {
        document.getElementById('status').textContent =
          online ? 'Online' : 'Offline';
      }
    </script>
  </head>
  <body ononline="update(true)"
        onoffline="update(false)"
        onload="update(navigator.onLine)">
    <p>You are: <span id="status">(Unknown)</span></p>
  </body>
</html>
```

#### 4.3.2 L' [article](#) élément



##### Catégories :

[Contenu du flux](#) .

[Sectionnement du contenu](#) .

[Contenu palpable](#) .

**Contextes dans lesquels cet élément peut être utilisé :**

Où [le contenu de sectionnement](#) est attendu.

**Modèle de contenu :**

[Contenu du flux](#) .

**Omission de balise dans text/html :**

Aucune des deux balises n'est omise.

**Attributs de contenu :**

[Attributs globaux](#)

**Considérations d'accessibilité :**

[Pour les auteurs](#) .

[Pour les exécutants](#) .

**Interface DOM :**

Utilisations [HTML<sup>Element</sup>](#).

L' [article](#)élément [représente](#) une composition complète ou autonome dans un document, une page, une application ou un site et qui est, en principe, distribuable ou réutilisable indépendamment, par exemple dans la syndication. Il peut s'agir d'un message sur un forum, d'un article de magazine ou de journal, d'une entrée de blog, d'un commentaire soumis par un utilisateur, d'un widget ou d'un gadget interactif ou de tout autre élément de contenu indépendant.

Lorsque [article](#)les éléments sont imbriqués, les éléments internes [article](#)représentent des articles qui sont en principe liés au contenu de l'article externe. Par exemple, une entrée de blog sur un site qui accepte les commentaires soumis par les utilisateurs peut représenter les commentaires comme [article](#)des éléments imbriqués dans l' [article](#)élément de l'entrée de blog.

Les informations d'auteur associées à un [article](#)élément (c'est-à-dire l' [address](#)élément) ne s'appliquent pas aux [article](#)éléments imbriqués.

*Lorsqu'il est utilisé spécifiquement avec du contenu à redistribuer dans la syndication, l' [article](#)élément a un objectif similaire à l' [entry](#)élément dans Atom. [\[ATOME\]](#)*

*Le vocabulaire de microdonnées [schema.org](#) peut être utilisé pour fournir la date de publication d'un [article](#)élément, en utilisant l'un des sous-types CreativeWork.*

Lorsque le contenu principal de la page (c'est-à-dire à l'exclusion des pieds de page, des en-têtes, des blocs de navigation et des barres latérales) est une seule composition autonome, ce contenu peut être marqué d'un , mais il est techniquement redondant dans ce cas (puisque l' [article](#)autonome évident que la page est une composition unique, car il s'agit d'un document unique).



Cet exemple montre un article de blog utilisant l' article élément, avec quelques annotations schema.org :

```
<article itemscope itemtype="http://schema.org/BlogPosting">
  <header>
    <h2 itemprop="headline">The Very First Rule of Life</h2>
    <p><time itemprop="datePublished" datetime="2009-10-09">3 days ago</time></p>
    <link itemprop="url" href="?comments=0">
  </header>
  <p>If there's a microphone anywhere near you, assume it's hot and sending whatever you're saying to the world. Seriously.</p>
  <p>...</p>
  <footer>
    <a itemprop="discussionUrl" href="?comments=1">Show comments...</a>
  </footer>
</article>
```

Voici ce même article de blog, mais montrant certains des commentaires :

```
<article itemscope itemtype="http://schema.org/BlogPosting">
  <header>
    <h2 itemprop="headline">The Very First Rule of Life</h2>
    <p><time itemprop="datePublished" datetime="2009-10-09">3 days ago</time></p>
    <link itemprop="url" href="?comments=0">
  </header>
  <p>If there's a microphone anywhere near you, assume it's hot and sending whatever you're saying to the world. Seriously.</p>
  <p>...</p>
  <section>
    <h1>Comments</h1>
    <article itemprop="comment" itemscope itemtype="http://schema.org/Comment" id="c1">
      <link itemprop="url" href="#c1">
      <footer>
        <p>Posted by: <span itemprop="creator" itemscope itemtype="http://schema.org/Person">
          <span itemprop="name">George Washington</span>
        </span></p>
        <p><time itemprop="dateCreated" datetime="2009-10-10">15 minutes ago</time></p>
      </footer>
    </article>
  </section>
</article>
```

```

    <p>Yeah! Especially when talking about your lobbyist
    friends!</p>
  </article>

  <article itemprop="comment" itemscope
    itemtype="http://schema.org/Comment" id="c2">

    <link itemprop="url" href="#c2">

    <footer>

      <p>Posted by: <span itemprop="creator" itemscope
        itemtype="http://schema.org/Person">

        <span itemprop="name">George Hammond</span>

      </span></p>

      <p><time itemprop="dateCreated" datetime="2009-10-10">5 minutes
      ago</time></p>

    </footer>

    <p>Hey, you have the same first name as me.</p>

  </article>
</section>
</article>

```

Remarquez l'utilisation de footer pour donner l'information pour chaque commentaire (comme qui l'a écrit et quand) : l' footer élément *peut* apparaître au début de sa section le cas échéant, comme dans ce cas. (Utiliser header dans ce cas ne serait pas mal non plus ; c'est surtout une question de préférence de création.)

Dans cet exemple, article les éléments sont utilisés pour héberger des widgets sur une page de portail. Les widgets sont implémentés en tant qu'éléments intégrés personnalisés afin d'obtenir un style spécifique et un comportement de script.

```

<!DOCTYPE HTML>
<html lang=en>
<title>eHome Portal</title>
<script src="/scripts/widgets.js"></script>
<link rel=stylesheet href="/styles/main.css">
<article is="stock-widget">
  <h2>Stocks</h2>
  <table>
    <thead> <tr> <th> Stock <th> Value <th> Delta
    <tbody> <template> <tr> <td> <td> <td> </template>
  </table>
  <p> <input type=button value="Refresh"
  onclick="this.parentElement.refresh()" ">
</article>
<article is="news-widget">
  <h2>News</h2>

```

```

<ul>
  <template>
    <li>
      <p><img> <strong></strong>
    <p>
  </template>
</ul>

<p> <input type=button value="Refresh"
onclick="this.parentElement.refresh()" ">
</article>

```

### 4.3.3 L' **section**élément



#### Catégories :

[Contenu du flux](#) .  
[Sectionnement du contenu](#) .  
[Contenu palpable](#) .

#### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de sectionnement](#) est attendu.

#### Modèle de contenu :

[Contenu du flux](#) .

#### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

#### Attributs de contenu :

[Attributs globaux](#)

#### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

#### Interface DOM :

Utilisations [HTMLElement](#).

L' **section**élément [représente](#) une section générique d'un document ou d'une application. Une section, dans ce contexte, est un regroupement thématique de contenu, généralement avec un titre.

Des exemples de sections seraient des chapitres, les différentes pages à onglets dans une boîte de dialogue à onglets ou les sections numérotées d'une thèse. La

page d'accueil d'un site Web peut être divisée en sections pour une introduction, des articles d'actualité et des informations de contact.

*Les auteurs sont encouragés à utiliser l' article élément au lieu de l' section élément lorsqu'il serait judicieux de syndiquer le contenu de l'élément. L' section élément n'est pas un élément conteneur générique. Lorsqu'un élément n'est nécessaire qu'à des fins de style ou pour faciliter la création de scripts, les auteurs sont encouragés à utiliser l' div élément à la place. Une règle générale est que l' section élément n'est approprié que si le contenu de l'élément est répertorié explicitement dans le plan du document .*

Dans l'exemple suivant, nous voyons un article (partie d'une page Web plus grande) sur les pommes, contenant deux courtes sections.

```
<article>
  <hgroup>
    <h2>Apples</h2>
    <p>Tasty, delicious fruit!</p>
  </hgroup>
  <p>The apple is the pomaceous fruit of the apple tree.</p>
  <section>
    <h3>Red Delicious</h3>
    <p>These bright red apples are the most common found in many
      supermarkets.</p>
  </section>
  <section>
    <h3>Granny Smith</h3>
    <p>These juicy, green apples make a great filling for
      apple pies.</p>
  </section>
</article>
```

Voici un programme de fin d'études avec deux sections, une pour la liste des diplômés, et une pour la description de la cérémonie. (Le balisage de cet exemple présente un style inhabituel parfois utilisé pour minimiser la quantité d' espaces entre les éléments .)

```
<!DOCTYPE Html>
<Html Lang=En
  ><Head
    ><Title
      >Graduation Ceremony Summer 2022</Title
    ></Head
  ><Body
    ><H1
      >Graduation</H1
```

```

    ><Section
      ><H2
        >Ceremony</H2
      ><P
        >Opening Procession</P
      ><P
        >Speech by Valedictorian</P
      ><P
        >Speech by Class President</P
      ><P
        >Presentation of Diplomas</P
      ><P
        >Closing Speech by Headmaster</P
    ></Section
    ><Section
      ><H2
        >Graduates</H2
      ><Ul
        ><Li
          >Molly Carpenter</Li
        ><Li
          >Anastasia Luccio</Li
        ><Li
          >Ebenezar McCoy</Li
        ><Li
          >Karrin Murphy</Li
        ><Li
          >Thomas Raith</Li
        ><Li
          >Susan Rodriguez</Li
      ></Ul
    ></Section
  ></Body
</Html>

```

Dans cet exemple, un auteur de livre a balisé certaines sections en tant que chapitres et d'autres en tant qu'annexes, et utilise CSS pour styliser différemment les en-têtes dans ces deux classes de section.

```

<style>
  section { border: double medium; margin: 2em; }

```

```

    section.chapter h2 { font: 2em Roboto, Helvetica Neue, sans-serif;
}

    section.appendix h2 { font: small-caps 2em Roboto, Helvetica Neue,
sans-serif; }
</style>
<header>
    <hgroup>
        <h1>My Book</h1>
        <p>A sample with not much content</p>
    </hgroup>
    <p><small>Published by Dummy Publicorp Ltd.</small></p>
</header>
<section class="chapter">
    <h2>My First Chapter</h2>
    <p>This is the first of my chapters. It doesn't say much.</p>
    <p>But it has two paragraphs!</p>
</section>
<section class="chapter">
    <h2>It Continues: The Second Chapter</h2>
    <p>Bla dee bla, dee bla dee bla. Boom.</p>
</section>
<section class="chapter">
    <h2>Chapter Three: A Further Example</h2>
    <p>It's not like a battle between brightness and earthtones would
go
unnoticed.</p>
    <p>But it might ruin my story.</p>
</section>
<section class="appendix">
    <h2>Appendix A: Overview of Examples</h2>
    <p>These are demonstrations.</p>
</section>
<section class="appendix">
    <h2>Appendix B: Some Closing Remarks</h2>
    <p>Hopefully this long example shows that you <em>can</em> style
sections, so long as they are used to indicate actual
sections.</p>
</section>

```

#### 4.3.4 L' **nav**élément



## Catégories :

[Contenu du flux](#) .  
[Sectionnement du contenu](#) .  
[Contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de sectionnement](#) est attendu.

## Modèle de contenu :

[Contenu du flux](#) .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

[Attributs globaux](#)

## Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

## Interface DOM :

Utilisations [HTML<sup>E</sup>lement](#).

L' [nav](#)élément [représente](#) une section d'une page qui renvoie à d'autres pages ou à des parties de la page : une section avec des liens de navigation.

*Tous les groupes de liens d'une page n'ont pas besoin d'être dans un [nav](#)élément — l'élément est principalement destiné aux sections constituées de blocs de navigation majeurs. En particulier, il est courant que les pieds de page contiennent une courte liste de liens vers diverses pages d'un site, telles que les conditions d'utilisation, la page d'accueil et une page de copyright. L' [footer](#)élément seul est suffisant pour de tels cas ; alors qu'un [nav](#)élément peut être utilisé dans de tels cas, il est généralement inutile.*

*Les agents utilisateurs (tels que les lecteurs d'écran) qui ciblent les utilisateurs qui peuvent bénéficier de l'omission des informations de navigation dans le rendu initial, ou qui peuvent bénéficier de la disponibilité immédiate des informations de navigation, peuvent utiliser cet élément comme moyen de déterminer quel contenu sur le page à ignorer initialement ou à fournir sur demande (ou les deux).*

Dans l'exemple suivant, il y a deux [nav](#)éléments, un pour la navigation principale sur le site et un pour la navigation secondaire sur la page elle-même.

```
<body>
  <h1>The Wiki Center Of Exampland</h1>
  <nav>
    <ul>
      <li><a href="/">Home</a></li>
```

```

    <li><a href="/events">Current Events</a></li>
    ...more...
</ul>
</nav>
<article>
  <header>
    <h2>Demos in ExampLand</h2>
    <p>Written by A. N. Other.</p>
  </header>
  <nav>
    <ul>
      <li><a href="#public">Public demonstrations</a></li>
      <li><a href="#destroy">Demolitions</a></li>
      ...more...
    </ul>
  </nav>
  <div>
    <section id="public">
      <h2>Public demonstrations</h2>
      <p>...more...</p>
    </section>
    <section id="destroy">
      <h2>Demolitions</h2>
      <p>...more...</p>
    </section>
    ...more...
  </div>
  <footer>
    <p><a href="?edit">Edit</a> | <a href="?delete">Delete</a> | <a href="?Rename">Rename</a></p>
  </footer>
</article>
<footer>
  <p><small>© copyright 1998 ExampLand Emperor</small></p>
</footer>
</body>

```

Dans l'exemple suivant, la page comporte plusieurs endroits où des liens sont présents, mais un seul de ces endroits est considéré comme une section de navigation.

```

<body itemscope itemtype="http://schema.org/Blog">

```



```

<header>
  <h1>Wake up sheeple!</h1>
  <p><a href="news.html">News</a> -
    <a href="blog.html">Blog</a> -
    <a href="forums.html">Forums</a></p>
  <p>Last Modified: <span itemprop="dateModified">2009-04-
01</span></p>
  <nav>
    <h1>Navigation</h1>
    <ul>
      <li><a href="articles.html">Index of all articles</a></li>
      <li><a href="today.html">Things sheeple need to wake up for
today</a></li>
      <li><a href="successes.html">Sheeple we have managed to
wake</a></li>
    </ul>
  </nav>
</header>
<main>
  <article itemprop="blogPosts" itemscope
itemtype="http://schema.org/BlogPosting">
    <header>
      <h2 itemprop="headline">My Day at the Beach</h2>
    </header>
    <div itemprop="articleBody">
      <p>Today I went to the beach and had a lot of fun.</p>
      ...more content...
    </div>
    <footer>
      <p>Posted <time itemprop="datePublished" datetime="2009-10-
10">Thursday</time>.</p>
    </footer>
  </article>
  ...more blog posts...
</main>
<footer>
  <p>Copyright ©
    <span itemprop="copyrightYear">2010</span>
    <span itemprop="copyrightHolder">The Example Company</span>
  </p>
  <p><a href="about.html">About</a> -
    <a href="policy.html">Privacy Policy</a> -

```

```
<a href="contact.html">Contact Us</a></p>
</footer>
</body>
```

Vous pouvez également voir des annotations de microdonnées dans l'exemple ci-dessus qui utilisent le vocabulaire [schema.org](https://schema.org/) pour fournir la date de publication et d'autres métadonnées sur l'article de blog.

Un [nav](#) élément ne doit pas nécessairement contenir une liste, il peut également contenir d'autres types de contenu. Dans ce bloc de navigation, les liens sont fournis en langage clair :

```
<nav>
  <h1>Navigation</h1>
  <p>You are on my home page. To the north lies <a href="/blog">my
    blog</a>, from whence the sounds of battle can be heard. To the
    east
    you can see a large mountain, upon which many <a
    href="/school">school papers</a> are littered. Far up thus
    mountain
    you can spy a little figure who appears to be me, desperately
    scribbling a <a href="/school/thesis">thesis</a>.</p>
  <p>To the west are several exits. One fun-looking exit is labeled
  <a
  href="https://games.example.com/">"games"</a>. Another more
  boring-looking exit is labeled <a
  href="https://isp.example.net/">ISP™</a>.</p>
  <p>To the south lies a dark and dank <a href="/about">contacts
  page</a>. Cobwebs cover its disused entrance, and at one point you
  see a rat run quickly out of the page.</p>
</nav>
```

Dans cet exemple, [nav](#) est utilisé dans une application de messagerie, pour permettre à l'utilisateur de changer de dossier :

```
<p><input type="button" value="Compose" onclick="compose()"></p>
<nav>
  <h1>Folders</h1>
  <ul>
    <li> <a href="/inbox" onclick="return
    openFolder(this.href)">Inbox</a> <span class=count></span>
    <li> <a href="/sent" onclick="return
    openFolder(this.href)">Sent</a>
    <li> <a href="/drafts" onclick="return
    openFolder(this.href)">Drafts</a>
```

```

<li> <a href="/trash" onclick="return
openFolder(this.href)">Trash</a>

<li> <a href="/customers" onclick="return
openFolder(this.href)">Customers</a>

</ul>

</nav>

```

#### 4.3.5 L' **aside** élément



##### Catégories :

[Contenu du flux](#) .  
[Sectionnement du contenu](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de sectionnement](#) est attendu.

##### Modèle de contenu :

[Contenu du flux](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTMLElement](#).

L' **aside** élément [représente](#) une section d'une page constituée d'un contenu lié de manière tangentielle au contenu autour de l' **aside** élément et qui pourrait être considéré comme distinct de ce contenu. Ces sections sont souvent représentées sous forme de barres latérales dans la typographie imprimée.

L'élément peut être utilisé pour des effets typographiques tels que des guillemets ou des barres latérales, pour la publicité, pour des groupes d' [nav](#) éléments et pour d'autres contenus considérés comme distincts du contenu principal de la page.

*Il n'est pas approprié d'utiliser l' **aside** élément uniquement pour les parenthèses, car celles-ci font partie du flux principal du document.*

L'exemple suivant montre comment un `aparté` est utilisé pour baliser des informations générales sur la Suisse dans un reportage beaucoup plus long sur l'Europe.

```
<aside>
  <h2>Switzerland</h2>
  <p>Switzerland, a land-locked country in the middle of geographic
  Europe, has not joined the geopolitical European Union, though it
  is
  a signatory to a number of European treaties.</p>
</aside>
```

L'exemple suivant montre comment un `aparté` est utilisé pour marquer une citation tirée dans un article plus long.

```
...

<p>He later joined a large company, continuing on the same work.
<q>I love my job. People ask me what I do for fun when I'm not at
work. But I'm paid to do my hobby, so I never know what to
answer. Some people wonder what they would do if they didn't have to
work... but I know what I would do, because I was unemployed for a
year, and I filled that time doing exactly what I do now.</q></p>

<aside>
  <q>People ask me what I do for fun when I'm not at work. But I'm
  paid to do my hobby, so I never know what to answer.</q>
</aside>

<p>Of course his work – or should that be hobby? –
isn't his only passion. He also enjoys other pleasures.</p>

...
```

L'extrait suivant montre comment `aside` peut être utilisé pour les blogrolls et autres contenus annexes sur un blog :

```
<body>
  <header>
    <h1>My wonderful blog</h1>
    <p>My tagline</p>
  </header>
  <aside>
```

```

<!-- this aside contains two sections that are tangentially
related
to the page, namely, links to other blogs, and links to blog
posts
from this blog -->
<nav>
  <h2>My blogroll</h2>
  <ul>
    <li><a href="https://blog.example.com/">Example Blog</a>
  </ul>
</nav>
<nav>
  <h2>Archives</h2>
  <ol reversed>
    <li><a href="/last-post">My last post</a>
    <li><a href="/first-post">My first post</a>
  </ol>
</nav>
</aside>
<aside>
  <!-- this aside is tangentially related to the page also, it
contains twitter messages from the blog author -->
  <h1>Twitter Feed</h1>
  <blockquote cite="https://twitter.example.net/t31351234">
    I'm on vacation, writing my blog.
  </blockquote>
  <blockquote cite="https://twitter.example.net/t31219752">
    I'm going to go on vacation soon.
  </blockquote>
</aside>
<article>
  <!-- this is a blog post -->
  <h2>My last post</h2>
  <p>This is my last post.</p>
  <footer>
    <p><a href="/last-post" rel=bookmark>Permalink</a>
  </footer>
</article>
<article>
  <!-- this is also a blog post -->
  <h2>My first post</h2>

```

```

<p>This is my first post.</p>
<aside>
  <!-- this aside is about the blog post, since it's inside the
  <article> element; it would be wrong, for instance, to put the
  blogroll here, since the blogroll isn't really related to this
  post
  specifically, only to the page as a whole -->
  <h2>Posting</h2>
  <p>While I'm thinking about it, I wanted to say something about
  posting. Posting is fun!</p>
</aside>
<footer>
  <p><a href="/first-post" rel=bookmark>Permalink</a>
</footer>
</article>
<footer>
  <p><a href="/archives">Archives</a> -
  <a href="/about">About me</a> -
  <a href="/copyright">Copyright</a></p>
</footer>
</body>

```

#### 4.3.6 Les éléments **h1**, **h2**, **h3**, **h4**, **h5** et **h6**



##### Catégories :

[Contenu du flux](#) .  
[Contenu de l'en-tête](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un [hgroup](#) élément.  
 Où [le contenu du titre](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLHeadingElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  // also has obsolete members
```

```
};
```

Ces éléments [représentent](#) les titres de leurs sections.

La sémantique et la signification de ces éléments sont définies dans la section sur [les en-têtes et les contours](#) .

Ces éléments ont un [niveau d'en-tête](#) donné par le numéro dans leur nom. Le [niveau d'en-tête](#) correspond aux niveaux des sections imbriquées. L' [h1](#) élément est destiné à une section de niveau supérieur, [h2](#) à une sous-section, [h3](#) à une sous-sous-section, etc.

En ce qui concerne leurs plans de document respectifs (leurs structures d'en-tête et de section), ces deux extraits sont sémantiquement équivalents :

```
<body>
<h1>Let's call it a draw(ing surface)</h1>
<h2>Diving in</h2>
<h2>Simple shapes</h2>
<h2>Canvas coordinates</h2>
<h3>Canvas coordinates diagram</h3>
<h2>Paths</h2>
</body>
<body>
  <h1>Let's call it a draw(ing surface)</h1>
  <section>
    <h2>Diving in</h2>
  </section>
  <section>
    <h2>Simple shapes</h2>
  </section>
</body>
```

```

<section>
  <h2>Canvas coordinates</h2>
  <section>
    <h3>Canvas coordinates diagram</h3>
  </section>
</section>
<section>
  <h2>Paths</h2>
</section>
</body>

```

Les auteurs pourraient préférer le premier style pour sa concision, ou le second style pour ses crochets de style supplémentaires. Quel est le meilleur est purement une question de style de création préféré.

#### 4.3.7 L' **hgroup** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de l'en-tête](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du titre](#) est attendu.

##### Modèle de contenu :

Zéro ou plusieurs [p](#)éléments, suivis d'un élément [h1](#), [h2](#), [h3](#), [h4](#), [h5](#), ou [h6](#), suivis de zéro ou plusieurs [p](#)éléments, éventuellement mélangés avec [des éléments de support de script](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTMLElement](#).



L' [hgroup](#) élément [représente](#) un titre et un contenu associé. L'élément peut être utilisé pour regrouper un élément [h1](#)– [h6](#) avec un ou plusieurs [p](#) éléments contenant du contenu représentant un sous-titre, un titre alternatif ou un slogan.

Voici quelques exemples d'en-têtes valides contenus dans un [hgroup](#) élément.

```
<hgroup>
  <h1>The reality dysfunction</h1>
  <p>Space is not the only void</p>
</hgroup>
<hgroup>
  <h1>Dr. Strangelove</h1>
  <p>Or: How I Learned to Stop Worrying and Love the Bomb</p>
</hgroup>
```

#### 4.3.8 L' [header](#) élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de flux](#) est attendu.

##### Modèle de contenu :

[Contenu du flux](#) , mais sans descendants d'élément [header](#) ou [.footer](#)

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

S'il existe un élément [de contenu de sectionnement](#) ancêtre : [pour les auteurs](#) ; [pour les exécutants](#) .  
Sinon : [pour les auteurs](#) ; [pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML<sup>Element</sup>](#).

L' [header](#) élément [représente](#) un groupe d'aides d'introduction ou de navigation.

Un headerélément est censé contenir généralement un en-tête (un élément h1 - h6 ou un hgroupélément), mais ce n'est pas obligatoire. L'headerélément peut également être utilisé pour envelopper la table des matières d'une section, un formulaire de recherche ou tout logo pertinent.

Voici quelques exemples d'en-têtes. Ce premier est pour un jeu :

```
<header>
  <p>Welcome to...</p>
  <h1>Voidwars!</h1>
</header>
```

L'extrait de code suivant montre comment l'élément peut être utilisé pour baliser l'en-tête d'une spécification :

```
<header>
  <hgroup>
    <h1>Fullscreen API</h1>
    <p>Living Standard — Last Updated 19 October 2015</p>
  </hgroup>
  <dl>
    <dt>Participate:</dt>
    <dd><a href="https://github.com/whatwg/fullscreen">GitHub
whatwg/fullscreen</a></dd>
    <dt>Commits:</dt>
    <dd><a href="https://github.com/whatwg/fullscreen/commits">GitHub
whatwg/fullscreen/commits</a></dd>
  </dl>
</header>
```

L'headerélément ne sectionne pas le contenu ; il n'introduit pas de nouvelle section.

Dans cet exemple, la page a un titre de page donné par l'h1élément, et deux sous-sections dont les titres sont donnés par h2les éléments. Le contenu après l'headerélément fait toujours partie de la dernière sous-section commencée dans l'headerélément, car l'headerélément ne participe pas à l' algorithme de contour .

```
<body>
  <header>
    <h1>Little Green Guys With Guns</h1>
    <nav>
      <ul>
        <li><a href="/games">Games</a>
        <li><a href="/forum">Forum</a>
        <li><a href="/download">Download</a>
      </ul>
    </nav>
```

```

<h2>Important News</h2> <!-- this starts a second subsection -->
<!-- this is part of the subsection entitled "Important News" -->
<p>To play today's games you will need to update your client.</p>
<h2>Games</h2> <!-- this starts a third subsection -->
</header>
<p>You have three active games:</p>
<!-- this is still part of the subsection entitled "Games" -->
...

```

#### 4.3.9 L' **footer** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de flux](#) est attendu.

##### Modèle de contenu :

[Contenu du flux](#) , mais sans descendants d'élément [header](#) ou [.footer](#)

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

S'il existe un élément [de contenu de sectionnement](#) ancêtre : [pour les auteurs](#) ; [pour les exécutants](#) .  
 Sinon : [pour les auteurs](#) ; [pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTMLElement](#).

L' [footer](#) élément [représente](#) un pied de page pour son élément [de contenu de sectionnement](#) ancêtre le plus proche , ou pour [l'élément de corps](#) s'il n'existe pas un tel ancêtre. Un pied de page contient généralement des informations sur sa section, telles que son auteur, des liens vers des documents connexes, des données de copyright, etc.

Lorsque l' [footer](#) élément contient des sections entières, elles [représentent](#) des annexes, des index, de longs colophons, des accords de licence détaillés et d'autres contenus similaires.

*Les informations de contact de l'auteur ou de l'éditeur d'une section appartiennent à un address élément, éventuellement lui-même à l'intérieur d'un fichier footer. Les signatures et autres informations qui pourraient convenir à la fois à a header ou à a footer peuvent être placées dans l'un ou l'autre (ou ni l'un ni l'autre). L'objectif principal de ces éléments est simplement d'aider l'auteur à rédiger un balisage auto-explicatif facile à entretenir et à styliser. ils ne visent pas à imposer des structures spécifiques aux auteurs.*

Les pieds de page ne doivent pas nécessairement apparaître à la *fin* d'une section, bien qu'ils le fassent généralement.

Lorsqu'il n'y a pas d'élément de contenu de sectionnement ancêtre , cela s'applique à toute la page.

*L' footer élément n'est pas lui-même sectionnant le contenu ; il n'introduit pas de nouvelle section.*

Voici une page avec deux pieds de page, un en haut et un en bas, avec le même contenu :

```
<body>
  <footer><a href="..">Back to index...</a></footer>
  <hgroup>
    <h1>Lorem ipsum</h1>
    <p>The ipsum of all lorem</p>
  </hgroup>
  <p>A dolor sit amet, consectetur adipisicing elit, sed do eiusmod
tempor incididunt ut labore et dolore magna aliqua. Ut enim ad
minim
veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip
ex
ea commodo consequat. Duis aute irure dolor in reprehenderit in
voluptate velit esse cillum dolore eu fugiat nulla
pariatur. Excepteur sint occaecat cupidatat non proident, sunt in
culpa qui officia deserunt mollit anim id est laborum.</p>
  <footer><a href="..">Back to index...</a></footer>
</body>
```

Voici un exemple qui montre l' footer élément utilisé à la fois pour un pied de page à l'échelle du site et pour un pied de page de section.

```
<!DOCTYPE HTML>
<HTML LANG="en"><HEAD>
<TITLE>The Ramblings of a Scientist</TITLE>
<BODY>
<H1>The Ramblings of a Scientist</H1>
<ARTICLE>
```

```

<H1>Episode 15</H1>
<VIDEO SRC="/fm/015.ogv" CONTROLS PRELOAD>
  <P><A HREF="/fm/015.ogv">Download video</A>.</P>
</VIDEO>
<FOOTER> <!-- footer for article -->
  <P>Published <TIME DATETIME="2009-10-21T18:26-07:00">on
2009/10/21 at 6:26pm</TIME></P>
</FOOTER>
</ARTICLE>
<ARTICLE>
  <H1>My Favorite Trains</H1>
  <P>I love my trains. My favorite train of all time is a Köf.</P>
  <P>It is fun to see them pull some coal cars because they look so
dwarfed in comparison.</P>
  <FOOTER> <!-- footer for article -->
    <P>Published <TIME DATETIME="2009-09-15T14:54-07:00">on
2009/09/15 at 2:54pm</TIME></P>
  </FOOTER>
</ARTICLE>
<FOOTER> <!-- site wide footer -->
  <NAV>
    <P><A HREF="/credits.html">Credits</A> -
      <A HREF="/tos.html">Terms of Service</A> -
      <A HREF="/index.html">Blog Index</A></P>
  </NAV>
  <P>Copyright © 2009 Gordon Freeman</P>
</FOOTER>
</BODY>
</HTML>

```

Certaines conceptions de sites ont ce que l'on appelle parfois des "fat footers" - des pieds de page qui contiennent beaucoup de matériel, y compris des images, des liens vers d'autres articles, des liens vers des pages pour envoyer des commentaires, des offres spéciales... à certains égards, un tout " première page" dans le pied de page.

Ce fragment montre le bas d'une page sur un site avec un "fat footer":

```

...
<footer>
  <nav>
    <section>
      <h1>Articles</h1>

```

```

    <p> Go to the gym
with
    our somersaults class! Our teacher Jim takes you through the
paces
    in this two-part article. <a href="articles/somersaults/1">Part
1</a> · <a href="articles/somersaults/2">Part 2</a></p>
    <p> Tired of walking on the
edge of
    a cliff<!-- sic -->? Our guest writer Lara shows you how to
bumble
    your way through the bars. <a href="articles/kindplus/1">Read
more...</a></p>
    <p> The chips are down, now all
that's left is a potato. What can you do with it? <a
href="articles/crisps/1">Read more...</a></p>
</section>
<ul>
    <li> <a href="/about">About us...</a>
    <li> <a href="/feedback">Send feedback!</a>
    <li> <a href="/sitemap">Sitemap</a>
</ul>
</nav>
<p><small>Copyright © 2015 The Snacker —
    <a href="/tos">Terms of Service</a></small></p>
</footer>
</body>

```

#### 4.3.10 L' **address** élément



##### Catégories :

Contenu du flux .

Contenu palpable .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

##### Modèle de contenu :

Contenu de flux , mais sans descendants de contenu d'en-tête , sans descendants de contenu de sectionnement et sans descendants d'élément header , footer ou address

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

Utilisations HTMLElement.

L' addressélément représente les informations de contact de son ancêtre le plus proche article ou bodyélément. S'il s'agit de l'élément body , les informations de contact s'appliquent au document dans son ensemble.

Par exemple, une page du site Web du W3C relative au HTML peut inclure les informations de contact suivantes :

```
<ADDRESS>
  <A href="../People/Raggett/">Dave Raggett</A>,
  <A href="../People/Arnaud/">Arnaud Le Hors</A>,
  contact persons for the <A href="Activity">W3C HTML Activity</A>
</ADDRESS>
```

L' addressélément ne doit pas être utilisé pour représenter des adresses arbitraires (par exemple des adresses postales), à moins que ces adresses ne soient en fait les informations de contact pertinentes. (L' pélément est l'élément approprié pour marquer les adresses postales en général.)

L' addressélément ne doit pas contenir d'informations autres que les informations de contact.

Par exemple, ce qui suit est une utilisation non conforme de l' addressélément :

```
<ADDRESS>Last Modified: 1999/12/24 23:37:50</ADDRESS>
```

En règle générale, l' addressélément serait inclus avec d'autres informations dans un footerélément.

Les informations de contact pour un *nœud* nœud sont une collection d' addresséléments définis par la première entrée applicable de la liste suivante :

**Si le nœud est un articleélément**

**Si le nœud est un bodyélément**

Les informations de contact se composent de tous les [address](#) éléments qui ont *node* comme ancêtre et qui n'ont pas d'ancêtre d'un autre élément [body](#) ou d' [article](#) un élément descendant de *node* .

**Si le *nœud* a un élément ancêtre qui est un [article](#) élément**

**Si le *nœud* a un élément ancêtre qui est un [body](#) élément**

Les informations de contact du *nœud* sont les mêmes que les informations de contact de l'ancêtre le plus proche [article](#) ou [body](#) de l'élément, selon ce qui est le plus proche.

**Si [le document de nœud](#) du *nœud* a [un élément de corps](#)**

Les informations de contact du *nœud* sont les mêmes que les informations de contact de [l'élément body](#) du [Document](#).

**Sinon**

Il n'y a pas d'informations de contact pour *node* .

Les agents utilisateurs peuvent exposer les informations de contact d'un nœud à l'utilisateur, ou les utiliser à d'autres fins, telles que l'indexation des sections sur la base des informations de contact des sections.

Dans cet exemple, le pied de page contient des informations de contact et une notice de copyright.

```
<footer>
  <address>
    For more details, contact
    <a href="mailto:js@example.com">John Smith</a>.
  </address>
  <p><small>© copyright 2038 Example Corp.</small></p>
</footer>
```

### 4.3.11 Titres et contours

[h1](#)– [h6](#) les éléments ont un **niveau de titre** , qui est donné par le numéro dans le nom de l'élément.

Ces éléments [représentent des en-têtes](#) . Plus le [niveau de titre](#) d'un [titre](#) est bas, moins il y a de sections ancêtres dans le [titre](#) .

Le **plan** correspond à tous [les titres](#) d'un document, dans [l'ordre de l'arborescence](#) .



Le [plan](#) doit être utilisé pour générer des plans de document, par exemple lors de la génération de tables des matières. Lors de la création d'une table des matières interactive, les entrées doivent renvoyer l'utilisateur à l' [en-tête](#) approprié .

Si un document comporte un ou plusieurs [titres](#) , au moins un seul [titre](#) dans le [plan](#) doit avoir un [niveau de titre](#) de 1.

Chaque [titre](#) suivant une autre *piste* [de titre](#) dans le [plan](#) doit avoir un [niveau de titre](#) inférieur, égal ou supérieur de 1 au [niveau de titre](#) de la *piste* .

L'exemple suivant n'est pas conforme :

```
<body>
  <h4>Apples</h4>
  <p>Apples are fruit.</p>
  <section>
    <h2>Taste</h2>
    <p>They taste lovely.</p>
  </section>
</body>
```

Il pourrait s'écrire comme suit et serait alors conforme :

```
<body>
  <h1>Apples</h1>
  <p>Apples are fruit.</p>
  <section>
    <h2>Taste</h2>
    <p>They taste lovely.</p>
  </section>
</body>
```

#### 4.3.11.1 Exemples de contours

Le fragment de balisage suivant :

```
<body>
  <hgroup id="document-title">
    <h1>HTML: Living Standard</h1>
    <p>Last Updated 12 August 2016</p>
  </hgroup>
  <p>Some intro to the document.</p>
```

```
<h2>Table of contents</h2>
<ol id=toc>...</ol>
<h2>First section</h2>
<p>Some intro to the first section.</p>
</body>
```

... donne lieu à 3 titres de document :

1. <h1>HTML: Living Standard</h1>
2. <h2>Table of contents</h2>.
3. <h2>First section</h2>.

Une vue rendue du [contour](#) pourrait ressembler à :

Tout d'abord, voici un document, qui est un livre avec des chapitres et des sous-sections très courts :

```
<!DOCTYPE HTML>
<html lang=en>
<title>The Tax Book (all in one page)</title>
<h1>The Tax Book</h1>
<h2>Earning money</h2>
<p>Earning money is good.</p>
<h3>Getting a job</h3>
<p>To earn money you typically need a job.</p>
<h2>Spending money</h2>
<p>Spending is what money is mainly used for.</p>
<h3>Cheap things</h3>
<p>Buying cheap things often not cost-effective.</p>
<h3>Expensive things</h3>
<p>The most expensive thing is often not the most cost-effective
either.</p>
<h2>Investing money</h2>
<p>You can lend your money to other people.</p>
<h2>Losing money</h2>
<p>If you spend money or invest money, sooner or later you will
lose money.
<h3>Poor judgement</h3>
```

```
<p>Usually if you lose money it's because you made a mistake.</p>
```

Son [plan](#) pourrait être présenté comme suit :

1. Le livre des impôts
  1. Gagner de l'argent
    1. Obtenir un emploi
  2. Dépenser de l'argent
    1. Choses bon marché
    2. Choses chères
  3. Investir de l'argent
  4. Perdre de l'argent
    1. Mauvais jugement

Notez que l' [title](#) élément n'est pas un [titre](#) .

Un document peut contenir plusieurs en-têtes de niveau supérieur :

```
<!DOCTYPE HTML>
<html lang=en>
<title>Alphabetic Fruit</title>
<h1>Apples</h1>
<p>Pomaceous.</p>
<h1>Bananas</h1>
<p>Edible.</p>
<h1>Carambola</h1>
<p>Star.</p>
```

[Le plan](#) du document pourrait être présenté comme suit :

1. Pommes
2. Bananes
3. Carambole

[header](#) les éléments n'influencent pas le [plan](#) d'un document :

```
<!DOCTYPE HTML>
<html lang="en">
<title>We're adopting a child! - Ray's blog</title>
<h1>Ray's blog</h1>
<article>
  <header>
    <nav>
      <a href="?t=-1d">Yesterday</a>;
      <a href="?t=-7d">Last week</a>;
      <a href="?t=-1m">Last month</a>
```

```

</nav>
<h2>We're adopting a child!</h2>
</header>
<p>As of today, Janine and I have signed the papers to become
the proud parents of baby Diane! We've been looking forward to
this day for weeks.</p>
</article>
</html>

```

[Le plan](#) du document pourrait être présenté comme suit :

1. Le blog de Ray
  1. Nous adoptons un enfant !

---

L'exemple suivant est conforme, mais non encouragé car il n'a pas [de titre](#) dont [le niveau de titre](#) est 1 :

```

<!DOCTYPE HTML>
<html lang=en>
<title>Alphabetic Fruit</title>
<section>
  <h2>Apples</h2>
  <p>Pomaceous.</p>
</section>
<section>
  <h2>Bananas</h2>
  <p>Edible.</p>
</section>
<section>
  <h2>Carambola</h2>
  <p>Star.</p>
</section>

```

[Le plan](#) du document pourrait être présenté comme suit :

1.
  1. Pommes
  2. Bananes
  3. Carambole

L'exemple suivant est conforme, mais non recommandé car le [niveau de titre](#) du premier [titre](#) n'est pas 1 :

```
<!DOCTYPE HTML>
<html lang=en>
<title>Feathers on The Site of Encyclopedic Knowledge</title>
  <h2>A plea from our caretakers</h2>
  <p>Please, we beg of you, send help! We're stuck in the server
room!</p>
<h1>Feathers</h1>
<p>Epidermal growths.</p>
```

[Le plan](#) du document pourrait être présenté comme suit :

1.
  1. Un appel de nos gardiens
2. Plumes

#### 4.3.11.2 Exposer les contours aux utilisateurs

Les agents utilisateurs sont encouragés à exposer [les contours](#) des pages aux utilisateurs pour faciliter la navigation. Cela est particulièrement vrai pour les médias non visuels, par exemple les lecteurs d'écran.

Par exemple, un agent utilisateur pourrait mapper les touches fléchées comme suit :

**Shift + ← Left**

Aller à la rubrique précédente

**Shift + → Right**

Aller à la rubrique suivante

**Shift + ↑ Up**

Aller au titre suivant dont [le niveau](#) est un de moins que le niveau du titre actuel

**Shift + ↓ Down**

Aller au titre suivant dont [le niveau](#) est le même que le niveau du titre actuel

#### 4.3.12 Résumé d'utilisation

*Cette section est non normative.*

Élément	But Exemple
<u>body</u>	<p>Le contenu du document.</p> <pre>&lt;!DOCTYPE HTML&gt; &lt;html lang="en"&gt;   &lt;head&gt; &lt;title&gt;Steve Hill's Home Page&lt;/title&gt; &lt;/head&gt;   &lt;body&gt; &lt;p&gt;Hard Trance is My Life.&lt;/p&gt; &lt;/body&gt; &lt;/html&gt;</pre>
<u>article</u>	<p>Une composition complète ou autonome dans un document, une page, une application ou un site et qui est, en principe, distribuable ou réutilisable indépendamment, par exemple dans la syndication. Il peut s'agir d'un message sur un forum, d'un article de magazine ou de journal, d'une entrée de blog, d'un commentaire soumis par un utilisateur, d'un widget ou d'un gadget interactif ou de tout autre élément de contenu indépendant.</p> <pre>&lt;article&gt;   &lt;img src="/tumblr_masqy2s5ynlrzfqbp01_500.jpg" alt="Yellow smiley face with the caption 'masif'"&gt;   &lt;p&gt;My fave Masif tee so far!&lt;/p&gt;   &lt;footer&gt;Posted 2 days ago&lt;/footer&gt; &lt;/article&gt; &lt;article&gt;   &lt;img src="/tumblr_m9tf6wSr6Wlrzfqbp01_500.jpg" alt=""&gt;   &lt;p&gt;Happy 2nd birthday Masif Saturdays!!!&lt;/p&gt;   &lt;footer&gt;Posted 3 weeks ago&lt;/footer&gt; &lt;/article&gt;</pre>
<u>section</u>	<p>Section générique d'un document ou d'une application. Une section, dans ce contexte, est un regroupement thématique de contenu, généralement avec un titre.</p> <pre>&lt;h1&gt;Biography&lt;/h1&gt; &lt;section&gt;   &lt;h1&gt;The facts&lt;/h1&gt;   &lt;p&gt;1500+ shows, 14+ countries&lt;/p&gt; &lt;/section&gt; &lt;section&gt;   &lt;h1&gt;2010/2011 figures per year&lt;/h1&gt;   &lt;p&gt;100+ shows, 8+ countries&lt;/p&gt; &lt;/section&gt;</pre>
<u>nav</u>	<p>Une section d'une page qui renvoie à d'autres pages ou à des parties de la page : une section avec des liens de navigation.</p> <pre>&lt;nav&gt;   &lt;p&gt;&lt;a href="/"&gt;Home&lt;/a&gt;   &lt;p&gt;&lt;a href="/biog.html"&gt;Bio&lt;/a&gt;   &lt;p&gt;&lt;a href="/discog.html"&gt;Discog&lt;/a&gt; &lt;/nav&gt;</pre>
<u>aside</u>	<p>Une section d'une page constituée d'un contenu lié de manière tangentielle au contenu autour de l' <u>aside</u> élément et qui pourrait être considérée comme distincte de ce contenu. Ces sections sont souvent représentées sous forme de barres latérales dans la typographie imprimée.</p> <pre>&lt;h1&gt;Music&lt;/h1&gt;</pre>

Élément	But Exemple
	<pre> &lt;p&gt;As any burner can tell you, the event has a lot of trance.&lt;/p&gt; &lt;aside&gt;You can buy the music we played at our &lt;a href="buy.html"&gt;playlist page&lt;/a&gt;.&lt;/aside&gt; &lt;p&gt;This year we played a kind of trance that originated in Belgium, Germany, and the Netherlands in the mid-90s.&lt;/p&gt; </pre>
<u>h1–h6</u>	<p>Un en-tête</p> <pre> &lt;h1&gt;The Guide To Music On The Playa&lt;/h1&gt; &lt;h2&gt;The Main Stage&lt;/h2&gt; &lt;p&gt;If you want to play on a stage, you should bring one.&lt;/p&gt; &lt;h2&gt;Amplified Music&lt;/h2&gt; &lt;p&gt;Amplifiers up to 300W or 90dB are welcome.&lt;/p&gt; </pre>
<u>hgroup</u>	<p>Un titre et un contenu associé. L'élément peut être utilisé pour regrouper un élément <u>h1</u>– <u>h6</u> avec un ou plusieurs <u>p</u> éléments contenant du contenu représentant un sous-titre, un titre alternatif ou un slogan.</p> <pre> &lt;hgroup&gt;   &lt;h1&gt;Burning Music&lt;/h1&gt;   &lt;p&gt;The Guide To Music On The Playa&lt;/p&gt; &lt;/hgroup&gt; &lt;section&gt;   &lt;hgroup&gt;     &lt;h1&gt;Main Stage&lt;/h1&gt;     &lt;p&gt;The Fiction Of A Music Festival&lt;/p&gt;   &lt;/hgroup&gt;   &lt;p&gt;If you want to play on a stage, you should bring one.&lt;/p&gt; &lt;/section&gt; &lt;section&gt;   &lt;hgroup&gt;     &lt;h1&gt;Loudness!&lt;/h1&gt;     &lt;p&gt;Questions About Amplified Music&lt;/p&gt;   &lt;/hgroup&gt;   &lt;p&gt;Amplifiers up to 300W or 90dB are welcome.&lt;/p&gt; &lt;/section&gt; </pre>
<u>header</u>	<p>Un groupe d'aides d'introduction ou de navigation.</p> <pre> &lt;article&gt;   &lt;header&gt;     &lt;h1&gt;Hard Trance is My Life&lt;/h1&gt;     &lt;p&gt;By DJ Steve Hill and Technikal&lt;/p&gt;   &lt;/header&gt;   &lt;p&gt;The album with the amusing punctuation has red artwork.&lt;/p&gt; &lt;/article&gt; </pre>
<u>footer</u>	<p>Un pied de page pour son élément <a href="#">de contenu de sectionnement</a> ancêtre le plus proche , ou pour <a href="#">l'élément de corps</a> s'il n'y a pas un tel ancêtre. Un pied de page contient généralement des informations sur sa section, telles que son auteur, des liens vers des documents connexes, des données de copyright, etc.</p> <pre> &lt;article&gt;   &lt;h1&gt;Hard Trance is My Life&lt;/h1&gt;   &lt;p&gt;The album with the amusing punctuation has red artwork.&lt;/p&gt;   &lt;footer&gt;     &lt;p&gt;Artists: DJ Steve Hill and Technikal&lt;/p&gt;   &lt;/footer&gt; </pre>

Élément	But
	Exemple
	</article>

#### 4.3.12.1 Article ou rubrique ?

*Cette section est non normative.*

A [section](#) fait partie d'autre chose. Un [article](#) est sa propre chose. Mais comment savoir lequel est lequel ? Généralement, la vraie réponse est "cela dépend de l'intention de l'auteur".

Par exemple, on pourrait imaginer un livre avec un chapitre "Granny Smith" qui dit juste "Ces pommes vertes juteuses font une excellente garniture pour les tartes aux pommes."; ce serait [section](#) parce qu'il y aurait beaucoup d'autres chapitres sur (peut-être) d'autres types de pommes.

D'un autre côté, on pourrait imaginer un tweet ou un commentaire reddit ou un post tumblr ou une petite annonce dans un journal qui dirait simplement "Granny Smith. Ces pommes vertes juteuses font une excellente garniture pour les tartes aux pommes."; ce serait alors [articles](#) parce que c'était tout.

Un commentaire sur un article ne fait pas partie du [article](#) sur lequel il commente, il est donc le sien [article](#).

## 4.4 Regroupement de contenu

### 4.4.1 L'élément



#### Catégories :

[Contenu du flux](#) .

[Contenu palpable](#) .

#### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de flux](#) est attendu.

#### Modèle de contenu :

[Contenu de la phrase](#) .

#### Omission de balise dans text/html :



un `,,, ,addressarticleasideblockquotedetailsdivdlfieldset  
igcaptionfigurefooterformh1h2h3h4h5h6headerhgrouphrmainme  
nunavolppresectiontableulaaudiodelinsmapnoscript,`  
ou un `video` élément, ou un [élément personnalisé autonome](#) .

## Attributs globaux

Pour les auteurs .  
Pour les exécutants .

```
[Exposed=Window]

interface HTMLParagraphElement : HTMLElement {

    [HTMLConstructor] constructor();

    // also has obsolete members

};
```

Alors que les paragraphes sont généralement représentés dans les médias visuels par des blocs de texte qui sont physiquement séparés des blocs adjacents par des lignes vides, une feuille de style ou un agent utilisateur serait également justifié de présenter les sauts de paragraphe d'une manière différente, par exemple en utilisant des pilcrows en ligne (¶) .

Les exemples suivants sont des fragments HTML conformes :

```
<p>The little kitten gently seated herself on a piece of
carpet. Later in her life, this would be referred to as the time
the
cat sat on the mat.</p>

<fieldset>

  <legend>Personal information</legend>

  <p>

    <label>Name: <input name="n"></label>

    <label><input name="anon" type="checkbox"> Hide from other
users</label>
```

```

</p>
<p><label>Address: <textarea name="a"></textarea></label></p>
</fieldset>
<p>There was once an example from Femley,<br>
Whose markup was of dubious quality.<br>
The validator complained,<br>
So the author was pained,<br>
To move the error from the markup to the rhyming.</p>

```

L' pélément ne doit pas être utilisé lorsqu'un élément plus spécifique est plus approprié.

L'exemple suivant est techniquement correct :

```

<section>
  <!-- ... -->
  <p>Last modified: 2001-04-23</p>
  <p>Author: fred@example.com</p>
</section>

```

Cependant, il serait mieux balisé comme suit :

```

<section>
  <!-- ... -->
  <footer>Last modified: 2001-04-23</footer>
  <address>Author: fred@example.com</address>
</section>

```

Ou:

```

<section>
  <!-- ... -->
  <footer>
    <p>Last modified: 2001-04-23</p>
    <address>Author: fred@example.com</address>
  </footer>
</section>

```

*Les éléments de liste (en particulier les éléments ol et ul) ne peuvent pas être des enfants d' péléments. Lorsqu'une phrase contient une liste à puces, on peut donc se demander comment elle doit être balisée.*

*Par exemple, cette phrase fantastique a des puces relatives à*

- sorcières,

- voyage plus rapide que la lumière, et
- la télépathie,

et est discuté plus en détail ci-dessous.

La solution est de réaliser qu'un [paragraphe](#) , en termes HTML, n'est pas un concept logique, mais un concept structurel. Dans l'exemple fantastique ci-dessus, il y a en fait cinq [paragraphes](#) tels que définis par cette spécification : un avant la liste, un pour chaque puce et un après la liste.

Le balisage pour l'exemple ci-dessus pourrait donc être :

```
<p>For instance, this fantastic sentence has bullets relating to</p>
<ul>
  <li>wizards,
  <li>faster-than-light travel, and
  <li>telepathy,
</ul>
<p>and is further discussed below.</p>
```

Les auteurs souhaitant styliser de manière pratique de tels paragraphes « logiques » composés de plusieurs paragraphes « structurels » peuvent utiliser l'[div](#) élément au lieu de l'[p](#) élément .

Ainsi, par exemple, l'exemple ci-dessus pourrait devenir le suivant :

```
<div>For instance, this fantastic sentence has bullets relating to
<ul>
  <li>wizards,
  <li>faster-than-light travel, and
  <li>telepathy,
</ul>
and is further discussed below.</div>
```

Cet exemple a toujours cinq paragraphes structurels, mais maintenant l'auteur peut styliser uniquement le [div](#) au lieu d'avoir à considérer chaque partie de l'exemple séparément.

#### 4.4.2 L' [hr](#) élément



### Catégories :

Contenu du flux .

### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

### Modèle de contenu :

Rien .

### Omission de balise dans text/html :

Pas de balise de fin .

### Attributs de contenu :

Attributs globaux

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLHRElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  // also has obsolete members
```

```
};
```

L' hrélément représente une rupture thématique au niveau du paragraphe , par exemple un changement de scène dans une histoire ou une transition vers un autre sujet dans une section d'un ouvrage de référence.

L'extrait fictif suivant d'un manuel de projet montre deux sections qui utilisent l' hrélément pour séparer les sujets au sein de la section.

```
<section>
  <h1>Communication</h1>
  <p>There are various methods of communication. This section
  covers a few of the important ones used by the project.</p>
  <hr>
  <p>Communication stones seem to come in pairs and have mysterious
  properties:</p>
  <ul>
```

```

<li>They can transfer thoughts in two directions once activated
if used alone.</li>
<li>If used with another device, they can transfer one's
consciousness to another body.</li>
<li>If both stones are used with another device, the
consciousnesses switch bodies.</li>
</ul>
<hr>
<p>Radios use the electromagnetic spectrum in the meter range and
longer.</p>
<hr>
<p>Signal flares use the electromagnetic spectrum in the
nanometer range.</p>
</section>
<section>
<h1>Food</h1>
<p>All food at the project is rationed:</p>
<dl>
<dt>Potatoes</dt>
<dd>Two per day</dd>
<dt>Soup</dt>
<dd>One bowl per day</dd>
</dl>
<hr>
<p>Cooking is done by the chefs on a set rotation.</p>
</section>

```

Il n'y a pas besoin d'un hrélément entre les sections elles-mêmes, puisque les sectionéléments et les h1éléments impliquent eux-mêmes des changements thématiques.

L'extrait suivant de *Pandora's Star* de Peter F. Hamilton montre deux paragraphes qui précèdent un changement de scène et le paragraphe qui le suit. Le changement de scène, représenté dans le livre imprimé par un espace contenant une étoile solitaire centrée entre les deuxième et troisième paragraphes, est ici représenté à l'aide de l' hrélément .

```

<p>Dudley was ninety-two, in his second life, and fast approaching
time for another rejuvenation. Despite his body having the physical
age of a standard fifty-year-old, the prospect of a long degrading
campaign within academia was one he regarded with dread. For a
supposedly advanced civilization, the Intersolar Commonwealth could
be

```

```

appallingly backward at times, not to mention cruel.</p>
<p><i>Maybe it won't be that bad</i>, he told himself. The lie was
comforting enough to get him through the rest of the night's
shift.</p>
<hr>
<p>The Carlton AllLander drove Dudley home just after dawn. Like
the
astronomer, the vehicle was old and worn, but perfectly capable of
doing its job. It had a cheap diesel engine, common enough on a
semi-frontier world like Gralmond, although its drive array was a
thoroughly modern photoneural processor. With its high suspension
and
deep-tread tyres it could plough along the dirt track to the
observatory in all weather and seasons, including the metre-deep
snow
of Gralmond's winters.</p>

```

L' hr élément n'affecte pas le plan du document .

#### 4.4.3 L' **pre**élément



##### Catégories :

Contenu du flux .  
Contenu palpable .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

##### Modèle de contenu :

Contenu de la phrase .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

Attributs globaux

##### Considérations d'accessibilité :

Pour les auteurs .  
Pour les exécutants .

##### Interface DOM :

```
[Exposed=Window]

interface HTMLPreElement : HTMLElement {

    [HTMLConstructor] constructor();

    // also has obsolete members

};
```

L' preélément représente un bloc de texte préformaté, dans lequel la structure est représentée par des conventions typographiques plutôt que par des éléments.

*Dans la syntaxe HTML , un caractère de retour à la ligne précédant immédiatement la prebalise de début de l'élément est supprimé.*

Quelques exemples de cas où l' preélément pourrait être utilisé :

- Y compris un e-mail, avec des paragraphes indiqués par des lignes vides, des listes indiquées par des lignes précédées d'une puce, etc.
- Y compris des fragments de code informatique, avec une structure indiquée selon les conventions de ce langage.
- Affichage de l'art ASCII.

*Les auteurs sont encouragés à considérer comment le texte préformaté sera perçu lorsque la mise en forme est perdue, comme ce sera le cas pour les utilisateurs de synthétiseurs vocaux, d'afficheurs braille, etc. Pour des cas comme l'art ASCII, il est probable qu'une présentation alternative, telle qu'une description textuelle, serait plus universellement accessible aux lecteurs du document.*

Pour représenter un bloc de code informatique, l' preélément peut être utilisé avec un codeélément ; pour représenter un bloc de sortie d'ordinateur, l' preélément peut être utilisé avec un sampélément. De même, l' kbdélément peut être utilisé dans un preélément pour indiquer le texte que l'utilisateur doit saisir.

Cet élément a des exigences de rendu impliquant l'algorithme bidirectionnel .

Dans l'extrait suivant, un exemple de code informatique est présenté.

```
<p>This is the <code>Panel</code> constructor:</p>
<pre><code>function Panel(element, canClose, closeHandler) {
    this.element = element;
    this.canClose = canClose;
    this.closeHandler = function () { if (closeHandler)
closeHandler() };
}

```

```
}</code></pre>
```

Dans l'extrait suivant, les éléments `samp` et `kbd` sont mélangés dans le contenu d'un `pre` élément pour montrer une session de Zork I.

```
<pre><samp>You are in an open field west of a big white house with
a boarded
front door.
There is a small mailbox here.

></samp> <kbd>open mailbox</kbd>

<samp>Opening the mailbox reveals:
A leaflet.

></samp></pre>
```

Ce qui suit montre un poème contemporain qui utilise l' `pre` élément pour préserver sa mise en forme inhabituelle, qui fait partie intégrante du poème lui-même.

```
<pre>                                maxling

it is with a                    heart
                                heavy

that i admit loss of a feline
                                so      loved

a friend lost to the
                                unknown
                                (night)

~cdr 11dec07</pre>
```

#### 4.4.4 L' `blockquote` élément



**Catégories :**

[Contenu du flux](#) .

[Contenu palpable](#) .



### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

### Modèle de contenu :

Contenu du flux .

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

cite— Lien vers la source de la citation ou plus d'informations sur la modification

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLQuoteElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute USVString cite;
```

```
};
```

*L'HTMLQuoteElement interface est également utilisée par l'qélément.*

L' blockquote élément représente une section qui est citée à partir d'une autre source.

Le contenu à l'intérieur de a blockquote doit être cité d'une autre source, dont l'adresse, si elle en a une, peut être citée dans l' cite attribut.

Si l' cite attribut est présent, il doit s'agir d'une URL valide potentiellement entourée d'espaces . Pour obtenir le lien de citation correspondant, la valeur de l'attribut doit être analysée par rapport au nœud document de l'élément . Les agents utilisateurs peuvent autoriser les utilisateurs à suivre ces liens de citation, mais ils sont principalement destinés à un usage privé (par exemple, par des scripts côté serveur collectant des statistiques sur l'utilisation des citations par un site), et non aux lecteurs.

Le contenu d'un blockquote peut être abrégé ou peut avoir un contexte ajouté de la manière conventionnelle pour la langue du texte.

Par exemple, en anglais, cela se fait traditionnellement à l'aide de crochets. Considérez une page avec la phrase "Jane a mangé le cracker. Elle a ensuite dit qu'elle aimait les pommes et le poisson."; il pourrait être cité comme suit :

```
<blockquote>
  <p>[Jane] then said she liked [...] fish.</p>
</blockquote>
```

L'attribution de la citation, le cas échéant, doit être placée à l'extérieur de l' [blockquote](#) élément.

Par exemple, ici l'attribution est donnée dans un paragraphe après la citation :

```
<blockquote>
  <p>I contend that we are both atheists. I just believe in one
  fewer
  god than you do. When you understand why you dismiss all the other
  possible gods, you will understand why I dismiss yours.</p>
</blockquote>
<p>— Stephen Roberts</p>
```

Les autres exemples ci-dessous montrent d'autres façons de montrer l'attribution.

L' [cite](#) attribut IDL doit [refléter](#) l'attribut de contenu de l'élément `cite`.

Ici, un [blockquote](#) élément est utilisé en conjonction avec un [figure](#) élément et son [figcaption](#) pour relier clairement une citation à son attribution (qui ne fait pas partie de la citation et n'appartient donc pas à l'intérieur de lui- [blockquote](#) même):

```
<figure>
  <blockquote>
    <p>The truth may be puzzling. It may take some work to grapple
    with.
    It may be counterintuitive. It may contradict deeply held
    prejudices. It may not be consonant with what we desperately want
    to
    be true. But our preferences do not determine what's true. We
    have a
    method, and that method helps us to reach not absolute truth,
    only
    asymptotic approaches to the truth — never there, just closer
    and closer, always finding vast new oceans of undiscovered
    possibilities. Cleverly designed experiments are the key.</p>
  </blockquote>
  <figcaption>Carl Sagan, in "<cite>Wonder and Skepticism</cite>",
  from
```

```
the <cite>Skeptical Inquirer</cite> Volume 19, Issue 1 (January-February
1995)</figcaption>
</figure>
```

Cet exemple suivant montre l'utilisation de [cite](#) along [blockquote](#):

```
<p>His next piece was the aptly named <cite>Sonnet 130</cite>:</p>
<blockquote cite="https://quotes.example.org/s/sonnet130.html">
  <p>My mistress' eyes are nothing like the sun,<br>
  Coral is far more red, than her lips red,<br>
  ...
```

Cet exemple montre comment un message de forum pourrait utiliser [blockquote](#) pour montrer à quel message un utilisateur répond. L' [article](#) élément est utilisé pour chaque poste, pour marquer le filetage.

```
<article>
  <h1><a href="https://bacon.example.com/?blog=109431">Bacon on a
crowbar</a></h1>
  <article>
    <header><strong>t3yw</strong> 12 points 1 hour ago</header>
    <p>I bet a narwhal would love that.</p>
    <footer><a href="?pid=29578">permalink</a></footer>
  <article>
    <header><strong>greg</strong> 8 points 1 hour ago</header>
    <blockquote><p>I bet a narwhal would love that.</p></blockquote>
    <p>Dude narwhals don't eat bacon.</p>
    <footer><a href="?pid=29579">permalink</a></footer>
  <article>
    <header><strong>t3yw</strong> 15 points 1 hour ago</header>
    <blockquote>
      <blockquote><p>I bet a narwhal would love
that.</p></blockquote>
      <p>Dude narwhals don't eat bacon.</p>
    </blockquote>
    <p>Next thing you'll be saying they don't get capes and wizard
hats either!</p>
    <footer><a href="?pid=29580">permalink</a></footer>
  <article>
    <article>
      <header><strong>boing</strong> -5 points 1 hour ago</header>
      <p>narwhals are worse than ceiling cat</p>
      <footer><a href="?pid=29581">permalink</a></footer>
```

```

    </article>
  </article>
</article>
</article>
<article>
  <header><strong>fred</strong> 1 points 23 minutes ago</header>
  <blockquote><p>I bet a narwhal would love that.</p></blockquote>
  <p>I bet they'd love to peel a banana too.</p>
  <footer><a href="?pid=29582">permalink</a></footer>
</article>
</article>
</article>

```

Cet exemple montre l'utilisation d'un blockquote pour de courts extraits, démontrant qu'il n'est pas nécessaire d'utiliser p des éléments à l'intérieur blockquote des éléments :

```

<p>He began his list of "lessons" with the following:</p>
<blockquote>One should never assume that his side of
the issue will be recognized, let alone that it will
be conceded to have merits.</blockquote>
<p>He continued with a number of similar points, ending with:</p>
<blockquote>Finally, one should be prepared for the threat
of breakdown in negotiations at any given moment and not
be cowed by the possibility.</blockquote>
<p>We shall now discuss these points...

```

*Des exemples de la façon de représenter une conversation sont présentés dans une section ultérieure ; il n'est pas approprié d'utiliser les éléments cite et blockquote à cette fin.*

#### 4.4.5 L' **ol** élément

✓ MDN  
✓ MDN

##### Catégories :

Contenu du flux .

Si les enfants de l'élément incluent au moins un li élément : Contenu palpable .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

### Modèle de contenu :

Zéro ou plus li et éléments de support de script.

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

reversed — Numéroté la liste à l'envers

start — Valeur de départ de la liste

type — Sorte de marqueur de liste

### Considérations d'accessibilité :

Pour les auteurs.

Pour les exécutants.

### Interface DOM :

```
[Exposed=Window]

interface HTMLListElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute boolean reversed;

    [CEReactions] attribute long start;

    [CEReactions] attribute DOMString type;

    // also has obsolete members

};
```

L' ol élément représente une liste d'éléments, où les éléments ont été intentionnellement ordonnés, de sorte que la modification de l'ordre modifierait la signification du document.

Les éléments de la liste sont les li nœuds enfants de l' ol élément, dans l'ordre de l'arborescence.

L' **reversed**attribut est un [attribut booléen](#) . S'il est présent, il indique que la liste est une liste décroissante (... , 3, 2, 1). Si l'attribut est omis, la liste est une liste ascendante (1, 2, 3, ...).

L' **start**attribut, s'il est présent, doit être un [entier valide](#) . Il est utilisé pour déterminer la [valeur de départ](#) de la liste.

Un **ol**élément a une **valeur de départ** , qui est un nombre entier déterminé comme suit :

1. Si l' **ol**élément a un **start**attribut, alors :
  1. Soit *parsed* le résultat de [l'analyse de la valeur de l'attribut sous la forme d'un entier](#) .
  2. Si *parsed* n'est pas une erreur, retournez *parsed* .
2. Si l' **ol**élément a un **reversed** attribut, renvoie le nombre d' [éléments possédés](#)**li** .
3. Retour 1.

L' **type**attribut peut être utilisé pour spécifier le type de marqueur à utiliser dans la liste, dans les cas où cela importe (par exemple parce que les éléments doivent être [référéncés](#) par leur numéro/lettre). L'attribut, s'il est spécifié, doit avoir une valeur identique à l'un des caractères donnés dans la première cellule d'une des lignes du tableau suivant. L' **type**attribut représente l'état indiqué dans la cellule de la deuxième colonne de la ligne dont la première cellule correspond à la valeur de l'attribut ; si aucune des cellules ne correspond, ou si l'attribut est omis, alors l'attribut représente l'état [décimal](#) .

Mot-clé	État	Description	Exemples pour les valeurs 1-3 et 3999-4001							
<b>1</b> (U+0031)	décimal	Nombres décimaux	1.	2.	3.	..	3999.	4000	4001.	..
<b>a</b> (U+0061)	alpha inférieur	Alphabet latin minuscule	un	b.	c.	..	euh.	ewv.	eww.	..
<b>A</b> (U+0041)	alpha supérieur	Alphabet latin majuscule	UN	B	C	..	EWU.	EWV.	EWV.	..
<b>i</b> (U+0069)	bas-romain	Chiffres romains minuscules	je	ii	iii	..	mmcmxcix	i <sup>-</sup> v <sup>-</sup>	i <sup>-</sup> v <sup>-</sup> i	..
<b>I</b> (U+0049)	haut-romain	Chiffres romains majuscules	JE	II	III	..	MMCMXCIX	I <sup>-</sup> V <sup>-</sup>	I <sup>-</sup> V <sup>-</sup> I	..

Les agents utilisateurs doivent restituer les éléments de la liste d'une manière cohérente avec l'état de l' typeattribut de l' olélément. Les nombres inférieurs ou égaux à zéro doivent toujours utiliser le système décimal, quel que soit l' typeattribut.

*Pour les agents utilisateurs CSS, un mappage de cet attribut à la propriété CSS 'list-style-type' est donné dans la section Rendering (le mappage est simple : les états ci-dessus ont les mêmes noms que leurs valeurs CSS correspondantes). Il est possible de redéfinir les styles de liste CSS par défaut utilisés pour implémenter cet attribut dans les agents utilisateurs CSS ; cela affectera la façon dont les éléments de la liste sont rendus.*

Les attributs reversed et typeIDL doivent refléter les attributs de contenu respectifs du même nom.

L' startattribut IDL doit refléter l'attribut de contenu du même nom, avec une valeur par défaut de 1.

*Cela signifie que l' startattribut IDL ne correspond pas nécessairement à la valeur de départ de la liste , dans les cas où l' startattribut content est omis et que l' reversedattribut content est spécifié.*

Le balisage suivant montre une liste où l'ordre est important, et où l' ol élément est donc approprié. Comparez cette liste à la liste équivalente dans la ul section pour voir un exemple des mêmes éléments utilisant l' ulélément.

```
<p>I have lived in the following countries (given in the order of
when
I first lived there):</p>
<ol>
  <li>Switzerland
  <li>United Kingdom
  <li>United States
  <li>Norway
</ol>
```

Notez comment changer l'ordre de la liste change la signification du document. Dans l'exemple suivant, la modification de l'ordre relatif des deux premiers éléments a modifié le lieu de naissance de l'auteur :

```
<p>I have lived in the following countries (given in the order of
when
I first lived there):</p>
<ol>
  <li>United Kingdom
  <li>Switzerland
  <li>United States
```

```
<li>Norway  
</ol>
```

#### 4.4.6 L' `ul` élément



##### Catégories :

Contenu du flux .

Si les enfants de l'élément incluent au moins un `li` élément : Contenu palpable .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

##### Modèle de contenu :

Zéro ou plus `li` et éléments de support de script .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

Attributs globaux

##### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

##### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLUListElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  // also has obsolete members
```

```
};
```

L' `ul` élément représente une liste d'éléments, où l'ordre des éléments n'est pas important — c'est-à-dire, où le changement de l'ordre ne changerait pas matériellement la signification du document.



Les éléments de la liste sont les linoeuds enfants de l' ul élément.

Le balisage suivant montre une liste où l'ordre n'a pas d'importance, et où l' ulélément est donc approprié. Comparez cette liste à la liste équivalente dans la olsection pour voir un exemple des mêmes éléments utilisant l' ol élément.

```
<p>I have lived in the following countries:</p>
<ul>
  <li>Norway
  <li>Switzerland
  <li>United Kingdom
  <li>United States
</ul>
```

Notez que changer l'ordre de la liste ne change pas la signification du document. Les éléments de l'extrait ci-dessus sont donnés par ordre alphabétique, mais dans l'extrait ci-dessous, ils sont donnés par ordre de taille de leur solde de compte courant en 2007, sans changer le sens du document :

```
<p>I have lived in the following countries:</p>
<ul>
  <li>Switzerland
  <li>Norway
  <li>United Kingdom
  <li>United States
</ul>
```

#### 4.4.7 L' menuélément



##### Catégories :

Contenu du flux .

Si les enfants de l'élément incluent au moins un liélément : Contenu palpable .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

##### Modèle de contenu :

Zéro ou plus liet éléments de support de script .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLMenuElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  // also has obsolete members
```

```
};
```

L' menuélément représente une barre d'outils composée de son contenu, sous la forme d'une liste non ordonnée d'éléments (représentés par li des éléments), dont chacun représente une commande que l'utilisateur peut exécuter ou activer.

*L' menuélément est simplement une alternative sémantique pour ul exprimer une liste non ordonnée de commandes (une "barre d'outils").*

Dans cet exemple, une application d'édition de texte utilise un menuélément pour fournir une série de commandes d'édition :

```
<menu>
  <li><button onclick="copy()" "></button></li>
  <li><button onclick="cut()" "></button></li>
  <li><button onclick="paste()" "></button></li>
</menu>
```

Notez que le style pour que cela ressemble à un menu de barre d'outils conventionnel dépend de l'application.

#### 4.4.8 L' liélément



### Catégories :

Aucun.

### Contextes dans lesquels cet élément peut être utilisé :

À l'intérieur ol.

À l'intérieur ul.

Éléments intérieurs menu.

### Modèle de contenu :

Contenu du flux .

### Omission de balise dans text/html :

La balise de fin d'un li élément peut être omise si l' élément est immédiatement suivi d'un autre élément ou s'il n'y a plus de contenu dans l'élément parent. li- li

### Attributs de contenu :

Attributs globaux

Si l'élément n'est pas un enfant d'un élément ul ou menu: value — Valeur ordinale de l'élément de liste

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLLIElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute long value;
```

```
    // also has obsolete members
```

```
};
```

L' li élément représente un élément de liste. Si son élément parent est un élément ol, ul ou menu, alors l'élément est un élément de la liste de l'élément parent, tel que défini pour ces éléments. Sinon, l'élément de liste n'a pas de relation définie liée à la liste avec un autre li élément.

L' **value** attribut, s'il est présent, doit être un [entier valide](#) . Il est utilisé pour déterminer la [valeur ordinale](#) de l'élément de liste, lorsque le [propriétaire](#) [li](#) de la liste est un élément. [ol](#)

---

Tout élément dont [la valeur calculée](#) de '[display](#)' est 'list-item' a un **propriétaire de liste** , qui est déterminé comme suit :

1. Si l'élément n'est pas [rendu](#) , retournez null ; l'élément n'a pas [de propriétaire de liste](#) .
2. Soit *ancêtre* le parent de l'élément.
3. Si l'élément a un ancêtre [ol](#), [ul](#) ou [menu](#), définissez *ancêtre* sur l'élément ancêtre le plus proche.
4. Renvoie l'ancêtre inclusif le plus proche de *ancestor* qui produit une [boîte CSS](#) .

*Un tel élément existera toujours, car à tout le moins l' [élément document](#) produira toujours un [CSS box](#) .*

Pour déterminer la **valeur ordinale** de chaque élément appartenant à un *propriétaire* [de liste](#) donné , procédez comme suit :

1. Que je sois 1.
2. Si *propriétaire* est un [ol](#) élément, laissez *la numérotation* être [la valeur de départ](#) de *propriétaire* . Sinon, laissez *la numérotation* être 1.
3. *Boucle* : si *i* est supérieur au nombre d' [éléments de liste que possède le propriétaire](#) , alors retourne ; tous les [éléments de la liste appartenant](#) au *propriétaire* ont reçu [des valeurs ordinales](#) .
4. Soit *item* le *i* ième des [éléments de la liste appartenant](#) au *propriétaire* , dans l' [ordre arborescent](#) .
5. If *item* is an [li](#) element that has a [value](#) attribute, then:
  1. Let *parsed* be the result of [parsing the value of the attribute as an integer](#).
  2. If *parsed* is not an error, then set *numbering* to *parsed*.
6. The [ordinal value](#) of *item* is *numbering*.

7. If *owner* is an `ol` element, and *owner* has a `reversed` attribute, decrement *numbering* by 1; otherwise, increment *numbering* by 1.
  8. Increment *i* by 1.
  9. Go to the step labeled *loop*.
- 

The `value` IDL attribute must [reflect](#) the value of the `value` content attribute.

The element's `value` IDL attribute does not directly correspond to its [ordinal value](#); it simply [reflects](#) the content attribute. For example, given this list:

```
<ol>
  <li>Item 1
  <li value="3">Item 3
  <li>Item 4
</ol>
```

The [ordinal values](#) are 1, 3, and 4, whereas the `value` IDL attributes return 0, 3, 0 on getting.

The following example, the top ten movies are listed (in reverse order). Note the way the list is given a title by using a `figure` element and its `figcaption` element.

```
<figure>
  <figcaption>The top 10 movies of all time</figcaption>
  <ol>
    <li value="10"><cite>Josie and the Pussycats</cite>, 2001</li>
    <li value="9"><cite lang="sh">Црна мачка, бели мачор</cite>, 1998</li>
    <li value="8"><cite>A Bug's Life</cite>, 1998</li>
    <li value="7"><cite>Toy Story</cite>, 1995</li>
    <li value="6"><cite>Monsters, Inc</cite>, 2001</li>
    <li value="5"><cite>Cars</cite>, 2006</li>
    <li value="4"><cite>Toy Story 2</cite>, 1999</li>
    <li value="3"><cite>Finding Nemo</cite>, 2003</li>
    <li value="2"><cite>The Incredibles</cite>, 2004</li>
    <li value="1"><cite>Ratatouille</cite>, 2007</li>
  </ol>
</figure>
```

The markup could also be written as follows, using the reversed attribute on the ol element:

```
<figure>
  <figcaption>The top 10 movies of all time</figcaption>
  <ol reversed>
    <li><cite>Josie and the Pussycats</cite>, 2001</li>
    <li><cite lang="sh">Црна мачка, бели мачор</cite>, 1998</li>
    <li><cite>A Bug's Life</cite>, 1998</li>
    <li><cite>Toy Story</cite>, 1995</li>
    <li><cite>Monsters, Inc</cite>, 2001</li>
    <li><cite>Cars</cite>, 2006</li>
    <li><cite>Toy Story 2</cite>, 1999</li>
    <li><cite>Finding Nemo</cite>, 2003</li>
    <li><cite>The Incredibles</cite>, 2004</li>
    <li><cite>Ratatouille</cite>, 2007</li>
  </ol>
</figure>
```

*Bien qu'il soit conforme d'inclure des éléments de titre (par exemple h1) à l'intérieur li des éléments, cela ne transmet probablement pas la sémantique voulue par l'auteur. Un en-tête commence une nouvelle section, donc un en-tête dans une liste divise implicitement la liste en plusieurs sections.*

#### 4.4.9 L' ol élément



##### Catégories :

Contenu du flux .

Si les enfants de l'élément incluent au moins un groupe nom-valeur : Contenu palpable .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

##### Modèle de contenu :

Soit : zéro ou plusieurs groupes composés chacun d'un ou plusieurs dt éléments suivis d'un ou plusieurs dd éléments, éventuellement mélangés à des éléments de support de script .

Ou : un ou plusieurs div éléments, éventuellement mélangés à des éléments de support de script .

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]

interface HTMLDListElement : HTMLElement {

    [HTMLConstructor] constructor();

    // also has obsolete members

};
```

L' dlélément représente une liste d'associations composée de zéro ou plusieurs groupes nom-valeur (une liste de description). Un groupe nom-valeur se compose d'un ou plusieurs noms ( dtéléments, éventuellement en tant qu'enfants d'un divélément enfant) suivis d'une ou plusieurs valeurs ( ddéléments, éventuellement en tant qu'enfants d'un divélément enfant), en ignorant tous les nœuds autres que dtles ddéléments enfants, et dtet ddles éléments qui sont des enfants d' divéléments enfants. Dans un même dlélément, il ne doit pas y avoir plus d'un dtélément pour chaque nom.

Les groupes nom-valeur peuvent être des termes et des définitions, des rubriques et des valeurs de métadonnées, des questions et des réponses ou tout autre groupe de données nom-valeur.

Les valeurs au sein d'un groupe sont des alternatives ; plusieurs paragraphes faisant partie de la même valeur doivent tous être donnés dans le même ddélément.

L'ordre de la liste des groupes et des noms et valeurs au sein de chaque groupe peut être significatif.

Afin d'annoter des groupes avec des attributs de microdonnées ou d'autres attributs globaux qui s'appliquent à des groupes entiers, ou simplement à des fins de style, chaque groupe d'un dlélément peut être enveloppé dans un divélément. Cela ne change pas la sémantique de l' dlélément.

Les groupes nom-valeur d'un dlélément *dl* sont déterminés à l'aide de l'algorithme suivant. Un groupe nom-valeur a un nom (une liste d' dtéléments, initialement vide) et une valeur (une liste d' ddéléments, initialement vide).

1. Soit *groupes* une liste vide de groupes nom-valeur.
2. Soit *courant* un nouveau groupe nom-valeur.
3. Soit *vuDd* faux.
4. Que *l' enfant* soit le premier enfant de *dl* .
5. Soit *petit-enfant* nul.
6. Tant que *child* n'est pas nul :
  1. Si *enfant* est un divélément, alors :
    1. Laissez *le petit-enfant* être le premier enfant de *l'enfant* .
    2. Alors que *petit-enfant* n'est pas nul :
      1. Processus dtoudd pour *petit-enfant* .
      2. Définissez *petit - enfant* sur le frère suivant du *petit - enfant* .
  2. Sinon, process dtordd for *child* .
  3. Définissez *l' enfant* sur le prochain frère de *l' enfant* .
7. Si *courant* n'est pas vide, alors ajoutez *courant* aux *groupes* .
8. *Groupes* de retour .

Traiter **ou** pour un *nœud* *nœud* signifie suivre ces étapes :**dtdd**

1. Soit *groups* , *current* et *seenDd* les mêmes variables que celles du même nom dans l'algorithme qui a invoqué ces étapes.
2. Si *node* est un dtélément, alors :
  1. Si *seenDd* est true, alors ajoutez *current* à *groups* , définissez *current* sur un nouveau groupe nom-valeur et définissez *seenDd* sur false.
  2. Ajouter *le nœud* au nom *actuel* de .
3. Sinon, si *node* est un ddélément, alors ajoutez *node* à la valeur de *current* et définissez *seenDd* sur true.



Lorsqu'un groupe nom-valeur a une liste vide comme nom ou valeur, cela est souvent dû à l'utilisation accidentelle ddd'éléments à la place d' dtéléments et vice versa. Les vérificateurs de conformité peuvent repérer de telles erreurs et peuvent être en mesure de conseiller les auteurs sur la manière d'utiliser correctement le balisage.

Dans l'exemple suivant, une entrée ("Auteurs") est liée à deux valeurs ("Jean" et "Luc").

```
<dl>
  <dt> Authors
  <dd> John
  <dd> Luke
  <dt> Editor
  <dd> Frank
</dl>
```

Dans l'exemple suivant, une définition est liée à deux termes.

```
<dl>
  <dt lang="en-US"> <dfn>color</dfn> </dt>
  <dt lang="en-GB"> <dfn>colour</dfn> </dt>
  <dd> A sensation which (in humans) derives from the ability of
    the fine structure of the eye to distinguish three differently
    filtered analyses of a view. </dd>
</dl>
```

L'exemple suivant illustre l'utilisation de l' dlélément pour baliser des métadonnées de sortes. À la fin de l'exemple, un groupe a deux étiquettes de métadonnées ("Auteurs" et "Editeurs") et deux valeurs ("Robert Rothman" et "Daniel Jackson"). Cet exemple utilise également l' divélément autour des groupes de dtet ddde l'élément, pour faciliter le style.

```
<dl>
  <div>
    <dt> Last modified time </dt>
    <dd> 2004-12-23T23:33Z </dd>
  </div>
  <div>
    <dt> Recommended update interval </dt>
    <dd> 60s </dd>
  </div>
  <div>
    <dt> Authors </dt>
    <dt> Editors </dt>
    <dd> Robert Rothman </dd>
  </div>
</dl>
```

```

    <dd> Daniel Jackson </dd>
  </div>
</dl>

```

L'exemple suivant montre l' [dl](#) élément utilisé pour donner un ensemble d'instructions. L'ordre des instructions ici est important (dans les autres exemples, l'ordre des blocs n'était pas important).

```

<p>Determine the victory points as follows (use the
first matching case):</p>
<dl>
  <dt> If you have exactly five gold coins </dt>
  <dd> You get five victory points </dd>
  <dt> If you have one or more gold coins, and you have one or more
silver coins </dt>
  <dd> You get two victory points </dd>
  <dt> If you have one or more silver coins </dt>
  <dd> You get one victory point </dd>
  <dt> Otherwise </dt>
  <dd> You get no victory points </dd>
</dl>

```

L'extrait de code suivant montre un [dl](#) élément utilisé comme glossaire. Notez l'utilisation de [dfn](#) pour indiquer le mot défini.

```

<dl>
  <dt><dfn>Apartment</dfn>, n.</dt>
  <dd>An execution context grouping one or more threads with one or
more COM objects.</dd>
  <dt><dfn>Flat</dfn>, n.</dt>
  <dd>A deflated tire.</dd>
  <dt><dfn>Home</dfn>, n.</dt>
  <dd>The user's login directory.</dd>
</dl>

```

Cet exemple utilise des attributs [de microdonnées](#) dans un [dl](#) élément, avec l' [div](#) élément, pour annoter les desserts glacés d'un restaurant français.

```

<dl>
  <div itemscope itemtype="http://schema.org/Product">
    <dt itemprop="name">Café ou Chocolat Liégeois
    <dd itemprop="offers" itemscope
itemtype="http://schema.org/Offer">
      <span itemprop="price">3.50</span>
      <data itemprop="priceCurrency" value="EUR">€</data>

```

```

<dd itemprop="description">
    2 boules Café ou Chocolat, 1 boule Vanille, sauce café ou
    chocolat, chantilly
</div>

<div itemscope itemtype="http://schema.org/Product">
    <dt itemprop="name">Américaine
    <dd itemprop="offers" itemscope
    itemtype="http://schema.org/Offer">
        <span itemprop="price">3.50</span>
        <data itemprop="priceCurrency" value="EUR">€</data>
        <dd itemprop="description">
            1 boule Crème brûlée, 1 boule Vanille, 1 boule Caramel,
            chantilly
        </div>
    </div>
</dl>

```

Sans l' div élément, le balisage devrait utiliser l' itemref attribut pour lier les données des dd éléments à l'élément, comme suit.

```

<dl>
    <dt itemscope itemtype="http://schema.org/Product" itemref="1-
    offer 1-description">
        <span itemprop="name">Café ou Chocolat Liégeois</span>
        <dd id="1-offer" itemprop="offers" itemscope
        itemtype="http://schema.org/Offer">
            <span itemprop="price">3.50</span>
            <data itemprop="priceCurrency" value="EUR">€</data>
            <dd id="1-description" itemprop="description">
                2 boules Café ou Chocolat, 1 boule Vanille, sauce café ou
                chocolat, chantilly
            </dd>
        </dd>
    <dt itemscope itemtype="http://schema.org/Product" itemref="2-
    offer 2-description">
        <span itemprop="name">Américaine</span>
        <dd id="2-offer" itemprop="offers" itemscope
        itemtype="http://schema.org/Offer">
            <span itemprop="price">3.50</span>
            <data itemprop="priceCurrency" value="EUR">€</data>
            <dd id="2-description" itemprop="description">
                1 boule Crème brûlée, 1 boule Vanille, 1 boule Caramel, chantilly
            </dd>
        </dd>
    </dt>
</dl>

```

*L' dl élément est inapproprié pour baliser un dialogue. Voir quelques exemples de balises de dialogue .*

#### 4.4.10 L' **dt**élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

Avant **dd** ou **dt** éléments à l'intérieur **dl** d'éléments.

Avant **dd** ou **dt** éléments à l'intérieur **div** d'éléments enfants d'un **dl** élément.

##### Modèle de contenu :

Contenu de flux , mais sans header, footer, contenu de sectionnement ou descendants de contenu d'en-tête .

##### Omission de balise dans text/html :

La balise de fin**dt** d'un élément peut être omise si l' élément est immédiatement suivi d'un autre élément ou d'un élément.**dt****dt****dd**

##### Attributs de contenu :

Attributs globaux

##### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

##### Interface DOM :

Utilisations HTML**Element**.

L' **dt**élément représente le terme, ou le nom, faisant partie d'un groupe de description de terme dans une liste de description ( **dl**élément).

*L' **dt**élément lui-même, lorsqu'il est utilisé dans un **dl**élément, n'indique pas que son contenu est un terme en cours de définition, mais cela peut être indiqué à l'aide de l' **dfn**élément.*

Cet exemple montre une liste de questions fréquemment posées (une FAQ) balisée à l'aide de l' **dt**élément pour les questions et de l' **dd**élément pour les réponses.

```
<article>
  <h1>FAQ</h1>
  <dl>
    <dt>What do we want?</dt>
    <dd>Our data.</dd>
    <dt>When do we want it?</dt>
    <dd>Now.</dd>
    <dt>Where is it?</dt>
    <dd>We are not sure.</dd>
  </dl>
```

```
</article>
```

#### 4.4.11 L' **dd**élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

Après dt ou dd éléments à l'intérieur dl des éléments.

Après dt ou dd éléments à l'intérieur div d'éléments enfants d'un dl élément.

##### Modèle de contenu :

Contenu du flux .

##### Omission de balise dans text/html :

La balise de fin dd d'un élément peut être omise si l' élément est immédiatement suivi d'un autre élément ou d'un élément, ou s'il n'y a plus de contenu dans l'élément parent. dddddt

##### Attributs de contenu :

Attributs globaux

##### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

##### Interface DOM :

Utilisations HTMLElement.

L' **dd**élément représente la description, la définition ou la valeur, faisant partie d'un groupe de termes-description dans une liste de description ( dlélément).

A dl peut être utilisé pour définir une liste de vocabulaire, comme dans un dictionnaire. Dans l'exemple suivant, chaque entrée, donnée par a dt avec a dfn, a plusieurs dds, montrant les différentes parties de la définition.

```
<dl>
  <dt><dfn>happiness</dfn></dt>
  <dd class="pronunciation">/'hæpɪnəs</dd>
  <dd class="part-of-speech"><i><abbr>n.</abbr></i></dd>
  <dd>The state of being happy.</dd>
  <dd>Good fortune; success. <q>Oh <b>happiness</b>! It
worked!</q></dd>
  <dt><dfn>rejoice</dfn></dt>
```

```

<dd class="pronunciation">/rɪˈdʒɔɪs/</dd>

<dd><i class="part-of-speech"><abbr>v.intr.</abbr></i> To be
delighted oneself.</dd>

<dd><i class="part-of-speech"><abbr>v.tr.</abbr></i> To cause one
to be delighted.</dd>
</dl>

```

#### 4.4.12 L' **figure** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de flux](#) est attendu.

##### Modèle de contenu :

Soit : un [figcaption](#) élément suivi du [contenu du flux](#) .  
 Ou : [contenu du flux](#) suivi d'un [figcaption](#) élément.  
 Ou : [contenu du flux](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML<sup>Element</sup>](#).

L' **figure** élément [représente](#) un [contenu de flux](#) , éventuellement avec une légende, qui est autonome (comme une phrase complète) et est généralement [référéncé](#) comme une seule unité du flux principal du document.

*"Autonome" dans ce contexte ne signifie pas nécessairement indépendant. Par exemple, chaque phrase d'un paragraphe est autonome ; une image qui fait partie d'une phrase serait inappropriée pour [figure](#), mais une phrase entière composée d'images conviendrait.*

L'élément peut ainsi être utilisé pour annoter des illustrations, des schémas, des photos, des listes de codes, etc.

Lorsqu'un [figure](#) est référencé à partir du contenu principal du document en l'identifiant par sa légende (par exemple, par un numéro de figure), il permet d'éloigner facilement ce contenu de ce contenu principal, par exemple, sur le côté de la page, pour pages dédiées, ou à une annexe, sans affecter le flux du document.

Si un [figure](#) élément est [référéncé](#) par sa position relative, par exemple, "dans la photographie ci-dessus" ou "comme le montre la figure suivante", alors déplacer la figure perturberait le sens de la page. Les auteurs sont encouragés à envisager d'utiliser des étiquettes pour faire référence aux figures, plutôt que d'utiliser de telles références relatives, de sorte que la page puisse facilement être restylée sans affecter le sens de la page.

Le premier [figcaption](#) élément enfant de l'élément, le cas échéant, représente la légende du [figure](#) contenu de l'élément. S'il n'y a pas [figcaption](#) d'élément enfant, il n'y a pas de légende.

Le [figure](#) contenu d'un élément fait partie du flux environnant. Si le but de la page est d'afficher la figure, par exemple une photographie sur un site de partage d'images, les éléments [figure](#) et [figcaption](#) peuvent être utilisés pour fournir explicitement une légende à cette figure. Pour le contenu qui n'est lié que de manière tangentielle, ou qui sert un objectif distinct du flux environnant, l'[aside](#) élément doit être utilisé (et peut lui-même envelopper un [figure](#)). Par exemple, une citation qui répète le contenu d'un [article](#) serait plus appropriée dans un [aside](#) que dans un [figure](#), car elle ne fait pas partie du contenu, c'est une répétition du contenu dans le but d'attirer les lecteurs ou de mettre en évidence des sujets clés.

Cet exemple montre l' [figure](#) élément pour baliser une liste de code.

```
<p>In <a href="#l4">listing 4</a> we see the primary core interface
API declaration.</p>
<figure id="l4">
  <figcaption>Listing 4. The primary core interface API
  declaration.</figcaption>
  <pre><code>interface PrimaryCore {
    boolean verifyDataLine();
    undefined sendData(sequence<byte> data);
    undefined initSelfDestruct();
  }</code></pre>
</figure>
<p>The API is designed to use UTF-8.</p>
```

On voit ici un [figure](#) élément pour baliser une photo qui est le contenu principal de la page (comme dans une galerie).

```
<!DOCTYPE HTML>
<html lang="en">
<title>Bubbles at work – My Gallery™</title>
```

```

<figure>
  
  <figcaption>Bubbles at work</figcaption>
</figure>

<nav><a href="19414.html">Prev</a> - <a
href="19416.html">Next</a></nav>

```

Dans cet exemple, nous voyons une image qui *n'est pas* une figure, ainsi qu'une image et une vidéo qui le sont. La première image fait littéralement partie de la deuxième phrase de l'exemple, ce n'est donc pas une unité autonome et figure serait donc inappropriée.

```

<h2>Malinko's comics</h2>

<p>This case centered on some sort of "intellectual property"
infringement related to a comic (see Exhibit A). The suit started
after a trailer ending with these words:

<blockquote>
  
</blockquote>

<p>...was aired. A lawyer, armed with a Bigger Notebook, launched a
preemptive strike using snowballs. A complete copy of the trailer
is
included with Exhibit B.

<figure>
  
  <figcaption>Exhibit A. The alleged <cite>rough copy</cite>
comic.</figcaption>
</figure>

<figure>
  <video src="ex-b.mov"></video>
  <figcaption>Exhibit B. The <cite>Rough Copy</cite>
trailer.</figcaption>
</figure>

<p>The case was resolved out of court.

```



Ici, une partie d'un poème est balisée à l'aide de [figure](#).

```
<figure>
  <p>'Twas brillig, and the slithy toves<br>
  Did gyre and gimble in the wabe;<br>
  All mimsy were the borogoves,<br>
  And the mome raths outgrabe.</p>
  <figcaption><cite>Jabberwocky</cite> (first verse). Lewis Carroll,
  1832-98</figcaption>
</figure>
```

Dans cet exemple, qui pourrait faire partie d'un travail beaucoup plus vaste sur un château, [figure](#) des éléments imbriqués sont utilisés pour fournir à la fois une légende de groupe et des légendes individuelles pour chaque personnage du groupe :

```
<figure>
  <figcaption>The castle through the ages: 1423, 1858, and 1999
  respectively.</figcaption>
  <figure>
    <figcaption>Etching. Anonymous, ca. 1423.</figcaption>
    
  </figure>
  <figure>
    <figcaption>Oil-based paint on canvas. Maria Towle,
    1858.</figcaption>
    
  </figure>
  <figure>
    <figcaption>Film photograph. Peter Jankle, 1999.</figcaption>
    
  </figure>
</figure>
```

L'exemple précédent pourrait également être écrit plus succinctement comme suit (en utilisant [title](#) des attributs à la place des paires [figure](#)/imbriquées [figcaption](#)) :

```
<figure>
  
  
```

```

    alt="The castle now has two towers and two walls.">
    
    <figcaption>The castle through the ages: 1423, 1858, and 1999
respectively.</figcaption>
</figure>

```

La figure n'est parfois [référéncée](#) qu'implicitement à partir du contenu :

```

<article>
  <h1>Fiscal negotiations stumble in Congress as deadline nears</h1>
  <figure>
    
    <figcaption>Barack Obama and Harry Reid. White House press
photograph.</figcaption>
  </figure>
  <p>Negotiations in Congress to end the fiscal impasse sputtered on
Tuesday, leaving both chambers
  grasping for a way to reopen the government and raise the
country's borrowing authority with a
  Thursday deadline drawing near.</p>
  ...
</article>

```

#### 4.4.13 L' **figcaption** élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

En tant que premier ou dernier enfant d'un [figure](#) élément.

##### Modèle de contenu :

[Contenu du flux](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .


[Pour les exécutants](#) .

### Interface DOM :

Utilisations [HTML<sup>E</sup>lement](#).

L' [figcaption](#)élément [représente](#) une légende ou une légende pour le reste du contenu de l' élément [figcaption](#)parent de l'élément [figure](#), le cas échéant.

L'élément peut contenir des informations supplémentaires sur la source :

```
<figcaption>
  <p>A duck.</p>
  <p><small>Photograph courtesy of  News.</small></p>
</figcaption>
<figcaption>
  <p>Average rent for 3-room apartments, excluding non-profit
apartments</p>
  <p>Zürich's Statistics Office — <time datetime=2017-11-14>14
November 2017</time></p>
</figcaption>
```

#### 4.4.14 L' [main](#)élément



### Catégories :

[Contenu du flux](#) .

[Contenu palpable](#) .

### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du flux](#) est attendu, mais uniquement s'il s'agit d'un [élément hiérarchiquement correct](#) [main](#) .

### Modèle de contenu :

[Contenu du flux](#) .

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

[Attributs globaux](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .

Pour les exécutants .

## Interface DOM :

Utilisations HTMLElement.

L' main élément représente le contenu dominant du document.

Un document ne doit pas avoir plus d'un main élément qui n'a pas l' hidden attribut spécifié.

Un **élément hiérarchiquement correct**main est un élément dont les éléments ancêtres sont limités à html, body, div, form sans nom accessible et des éléments personnalisés autonomes . Chaque main élément doit être un élément hiérarchiquement correctmain .

Dans cet exemple, l'auteur a utilisé une présentation où chaque composant de la page est rendu dans une boîte. Pour envelopper le contenu principal de la page (par opposition à l'en-tête, au pied de page, à la barre de navigation et à une barre latérale), l' main élément est utilisé.

```
<!DOCTYPE html>
<html lang="en">
<title>RPG System 17</title>
<style>
  header, nav, aside, main, footer {
    margin: 0.5em; border: thin solid; padding: 0.5em;
    background: #EFF; color: black; box-shadow: 0 0 0.25em #033;
  }
  h1, h2, p { margin: 0; }
  nav, main { float: left; }
  aside { float: right; }
  footer { clear: both; }
</style>
<header>
  <h1>System Eighteen</h1>
</header>
<nav>
  <a href=" ../16/">← System 17</a>
  <a href=" ../18/">RPXIX →</a>
</nav>
<aside>
  <p>This system has no HP mechanic, so there's no healing.
</aside>
<main>
```

```

<h2>Character creation</h2>

<p>Attributes (magic, strength, agility) are purchased at the cost
of one point per level.</p>

<h2>Rolls</h2>

<p>Each encounter, roll the dice for all your skills. If you roll
more than the opponent, you win.</p>

</main>

<footer>

  <p>Copyright © 2013

</footer>

</html>

```

Dans l'exemple suivant, plusieurs main éléments sont utilisés et un script est utilisé pour faire fonctionner la navigation sans aller-retour de serveur et pour définir l' hidden attribut sur ceux qui ne sont pas actuels :

```

<!doctype html>
<html lang=en-CA>
<meta charset=utf-8>
<title> ... </title>
<link rel=stylesheet href=spa.css>
<script src=spa.js async></script>
<nav>
  <a href=/>Home</a>
  <a href=/about>About</a>
  <a href=/contact>Contact</a>
</nav>
<main>
  <h1>Home</h1>
  ...
</main>
<main hidden>
  <h1>About</h1>
  ...
</main>
<main hidden>
  <h1>Contact</h1>
  ...
</main>

<footer>Made with ❤ by <a href=https://example.com/>Example
🐶</a>.</footer>

```

#### 4.4.15 L' **div**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le flux de contenu](#) est attendu.  
En tant qu'enfant d'un [dl](#)élément.

##### Modèle de contenu :

Si l'élément est un enfant d'un [dl](#)élément : un ou plusieurs [dt](#)éléments suivis d'un ou plusieurs [dd](#)éléments, éventuellement mélangés avec [des éléments de support de script](#) .  
Si l'élément n'est pas un enfant d'un [dl](#)élément : [flux content](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

```
[Exposed=Window]

interface HTMLDivElement : HTMLElement {

    [HTMLConstructor] constructor();

    // also has obsolete members

};
```

L' **div**élément n'a aucune signification particulière. Il [représente](#) ses enfants. Il peut être utilisé avec les attributs [class](#), [lang](#)et [title](#)pour baliser la sémantique commune à un groupe d'éléments consécutifs. Il peut également être utilisé dans un [dl](#) élément, en enveloppant des groupes d' éléments [dt](#)et [dd](#).

*Les auteurs sont fortement encouragés à considérer l' **div**élément comme un élément de dernier recours, lorsqu'aucun autre élément ne convient. L'utilisation*

*d'éléments plus appropriés au lieu de l' divélément conduit à une meilleure accessibilité pour les lecteurs et une maintenabilité plus facile pour les auteurs.*

Par exemple, un article de blog serait balisé à l'aide de article, un chapitre à l'aide de section, les aides à la navigation d'une page à l'aide nav et un groupe de contrôles de formulaire à l'aide de fieldset.

D'autre part, divles éléments peuvent être utiles à des fins stylistiques ou pour envelopper plusieurs paragraphes dans une section qui doivent tous être annotés de la même manière. Dans l'exemple suivant, nous voyons divdes éléments utilisés comme moyen de définir la langue de deux paragraphes à la fois, au lieu de définir la langue sur les deux éléments de paragraphe séparément :

```
<article lang="en-US">
  <h1>My use of language and my cats</h1>
  <p>My cat's behavior hasn't changed much since her absence, except
  that she plays her new physique to the neighbors regularly, in an
  attempt to get pets.</p>
  <div lang="en-GB">
    <p>My other cat, coloured black and white, is a sweetie. He
    followed
    us to the pool today, walking down the pavement with us.
    Yesterday
    he apparently visited our neighbours. I wonder if he recognises
    that
    their flat is a mirror image of ours.</p>
    <p>Hm, I just noticed that in the last paragraph I used British
    English. But I'm supposed to write in American English. So I
    shouldn't say "pavement" or "flat" or "colour"...</p>
  </div>
  <p>I should say "sidewalk" and "apartment" and "color"!</p>
</article>
```

## 4.5 Sémantique au niveau du texte

### 4.5.1 L' aélément

✓ MDN  
✓ MDN

#### Catégories :

Contenu du flux .

Contenu de la phrase .

Si l'élément a un hrefattribut : Contenu interactif .

Contenu palpable .

### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu du phrasé est attendu.

### Modèle de contenu :

Transparent , mais il ne doit y avoir aucun descendant de contenu interactif<sup>a</sup> , descendant d'élément ou descendant avec l' tabindexattribut spécifié.

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

href— Adresse du lien hypertexte

target— Navigable pour la navigation par lien hypertexte

download— S'il faut télécharger la ressource au lieu d'y accéder, et son nom de fichier si c'est le cas

ping— URL à pinger

rel— Relation entre l'emplacement dans le document contenant l' hyperlien et la ressource de destination

hreflang— Langue de la ressource liée

type— Indice pour le type de la ressource référencée

referrerpolicy— Politique de référence pour les récupérations initiées par l'élément

### Considérations d'accessibilité :

Si l'élément a un hrefattribut : pour les auteurs ; pour les exécutants .

Sinon : pour les auteurs ; pour les exécutants .

### Interface DOM :

[Exposed=Window]

```
interface HTMLAnchorElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute DOMString target;
```

```
    [CEReactions] attribute DOMString download;
```

```
    [CEReactions] attribute USVString ping;
```

```
    [CEReactions] attribute DOMString rel;
```



```

[SameObject, PutForwards=value] readonly attribute
DOMTokenList relList;

[CEReactions] attribute DOMString hreflang;

[CEReactions] attribute DOMString type;

[CEReactions] attribute DOMString text;

[CEReactions] attribute DOMString referrerPolicy;

// also has obsolete members

};

HTMLAnchorElement includes HTMLHyperlinkElementUtils;

```

Si l'[a](#)élément a un [href](#)attribut, alors il [représente](#) un [lien hypertexte](#) (une ancre hypertexte) étiqueté par son contenu.

Si l'[a](#)élément n'a pas [href](#)d'attribut, alors l'élément [représente](#) un espace réservé pour l'endroit où un lien aurait autrement été placé, s'il avait été pertinent, composé uniquement du contenu de l'élément.

Les attributs [target](#), [download](#), [ping](#), [rel](#), [hreflang](#), [type](#) et [referrerPolicy](#) doivent être omis si l'attribut [href](#) n'est pas présent.

Si l'[itemprop](#)attribut est spécifié sur un [a](#)élément, alors l'[href](#)attribut doit également être spécifié.

Si un site utilise une barre d'outils de navigation cohérente sur chaque page, le lien qui renverrait normalement à la page elle-même pourrait être balisé à l'aide d'un [a](#)élément :

```

<nav>
  <ul>
    <li> <a href="/">Home</a> </li>
    <li> <a href="/news">News</a> </li>
    <li> <a>Examples</a> </li>
    <li> <a href="/legal">Legal</a> </li>
  </ul>

```

```
</nav>
```

Les attributs [href](#), [target](#), [download](#), [ping](#) et [referrerpolicy](#) affectent ce qui se passe lorsque les utilisateurs [suivent des hyperliens](#) ou [téléchargent des hyperliens](#) créés à l'aide de l' [a](#) élément. Les attributs [rel](#), [hreflang](#) et [type](#) peuvent être utilisés pour indiquer à l'utilisateur la nature probable de la ressource cible avant que l'utilisateur ne suive le lien.

Le [comportement d'activation](#) d'un [élément a](#) element compte tenu d'un [événement](#) event est :

1. Si l'élément n'a pas [href](#) d'attribut, alors retournez.
2. Laissez *hyperlinkSuffix* être nul.
3. Si [la cible](#) de l' [événement](#) est un avec un attribut spécifié, alors [:imgismap](#)
  1. Soient x et y 0.
  2. Si l'attribut de l' [événement](#) [isTrusted](#) est initialisé à true, définissez x sur la distance en [pixels CSS](#) entre le bord gauche de l'image et l'emplacement du clic, et définissez y sur la distance en [pixels CSS](#) entre le bord supérieur de l'image et l'emplacement du clic.
  3. Si x est négatif, définissez x sur 0.
  4. Si y est négatif, réglez y sur 0.
  5. Définissez *hyperlinkSuffix* sur la concaténation de U+003F (?), la valeur de x exprimée sous forme d'entier de base dix à l'aide [de chiffres ASCII](#) , U+002C (,) et la valeur de y exprimée sous forme d'entier de base dix à l'aide de [chiffres ASCII](#) .
4. Si *element* a un [download](#) attribut, ou si l'utilisateur a exprimé une préférence pour télécharger le lien hypertexte, alors [téléchargez le lien hypertexte](#) créé par *element* donné *hyperlinkSuffix* .
5. Sinon, [suivez le lien hypertexte](#) créé par l'élément donné *hyperlinkSuffix* .

#### **a. [text](#)**

Identique à [textContent](#).



Les attributs IDL [download](#), [ping](#), [target](#), [rel](#), [hreflang](#) et [type](#), doivent [refléter](#) les attributs de contenu respectifs du même nom.



L'attribut IDL [relList](#) doit [refléter](#) l' [rel](#) attribut content.



L'attribut IDL `referrerPolicy` doit [réfléter](#) l' `referrerpolicy` attribut content, [limité aux seules valeurs connues](#) .

Le `text` getter de l'attribut doit renvoyer [le contenu textuel descendant](#) de cet élément .

Le `text` setter de l'attribut doit [chaîne remplacer all](#) par la valeur donnée dans cet élément.

L' `a` élément peut être enroulé autour de paragraphes entiers, de listes, de tableaux, etc., voire de sections entières, tant qu'il n'y a pas de contenu interactif à l'intérieur (par exemple, des boutons ou d'autres liens). Cet exemple montre comment cela peut être utilisé pour transformer un bloc publicitaire entier en lien :

```
<aside class="advertising">
  <h1>Advertising</h1>
  <a href="https://ad.example.com/?adid=1929&pubid=1422">
    <section>
      <h1>Mellblomatic 9000!</h1>
      <p>Turn all your widgets into mellbloms!</p>
      <p>Only $9.99 plus shipping and handling.</p>
    </section>
  </a>
  <a href="https://ad.example.com/?adid=375&pubid=1422">
    <section>
      <h1>The Mellblom Browser</h1>
      <p>Web browsing at the speed of light.</p>
      <p>No other browser goes faster!</p>
    </section>
  </a>
</aside>
```

L'exemple suivant montre comment un peu de script peut être utilisé pour transformer efficacement une ligne entière dans un tableau de liste d'emplois en lien hypertexte :

```
<table>
  <tr>
    <th>Position
    <th>Team
    <th>Location
  <tr>
```

```

<td><a href="/jobs/manager">Manager</a>
<td>Remotees
<td>Remote
<tr>
<td><a href="/jobs/director">Director</a>
<td>Remotees
<td>Remote
<tr>
<td><a href="/jobs/astronaut">Astronaut</a>
<td>Architecture
<td>Remote
</table>
<script>
document.querySelector("table").onclick = ({ target }) => {
  if (target.parentElement.localName === "tr") {
    const link = target.parentElement.querySelector("a");
    if (link) {
      link.click();
    }
  }
}
</script>

```

#### 4.5.2 L' **em** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

## Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

## Interface DOM :

Utilisations HTMLElement.

L' emélément représente l'accentuation de son contenu.

Le niveau de stress d'un contenu particulier est donné par son nombre d' eméléments ancêtres.

Le placement de l'emphase change le sens de la phrase. L'élément fait ainsi partie intégrante du contenu. La manière précise dont l'accent est utilisé de cette manière dépend de la langue.

Ces exemples montrent comment la modification de l'accentuation de l'accent modifie le sens. Tout d'abord, un constat général, sans accent :

```
<p>Cats are cute animals.</p>
```

En mettant l'accent sur le premier mot, la déclaration implique que le type d'animal dont il est question est en question (peut-être que quelqu'un affirme que les chiens sont mignons) :

```
<p><em>Cats</em> are cute animals.</p>
```

En déplaçant l'accent sur le verbe, on souligne que la vérité de toute la phrase est en cause (peut-être que quelqu'un dit que les chats ne sont pas mignons) :

```
<p>Cats <em>are</em> cute animals.</p>
```

En le déplaçant vers l'adjectif, la nature exacte des chats est réaffirmée (peut-être que quelqu'un a suggéré que les chats étaient des animaux *méchants*) :

```
<p>Cats are <em>cute</em> animals.</p>
```

De même, si quelqu'un affirmait que les chats étaient des légumes, quelqu'un corrigeant cela pourrait mettre l'accent sur le dernier mot :

```
<p>Cats are cute <em>animals</em>.</p>
```

En mettant l'accent sur toute la phrase, il devient clair que l'orateur se bat pour faire passer le message. Ce type d'accentuation affecte également généralement la ponctuation, d'où le point d'exclamation ici.

```
<p><em>Cats are cute animals!</em></p>
```

La colère mélangée à l'accent mis sur la gentillesse pourrait conduire à un balisage tel que :

```
<p><em>Cats are <em>cute</em> animals!</em></p>
```

L' emélément n'est pas un élément "italique" générique. Parfois, le texte est destiné à se démarquer du reste du paragraphe, comme s'il était d'une humeur ou d'une voix différente. Pour cela, l' iélément est plus approprié.

L' emélément n'est pas non plus destiné à transmettre de l'importance; à cette fin, l' strongélément est plus approprié.

#### 4.5.3 L' strongélément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML<sup>Element</sup>](#).

L' strongélément [représente](#) une importance, une gravité ou une urgence fortes pour son contenu.

**Importance :** l' strongélément peut être utilisé dans un titre, une légende ou un paragraphe pour distinguer la partie qui compte vraiment des autres parties qui pourraient être plus détaillées, plus joviales ou simplement passe-partout. (Ceci est distinct du balisage des sous-titres, pour lequel l' hgroupélément est approprié.)

Par exemple, le premier mot du paragraphe précédent est marqué par strong pour le distinguer du texte plus détaillé du reste du paragraphe.

**Gravité** : l' strongélément peut être utilisé pour baliser un avertissement ou une mise en garde.

**Urgence** : l' strongélément peut être utilisé pour désigner le contenu que l'utilisateur a besoin de voir plus tôt que les autres parties du document.

Le niveau d'importance relative d'un contenu est donné par son nombre d' strongéléments ancêtres ; chaque strongélément augmente l'importance de son contenu.

Changer l'importance d'un morceau de texte avec l' strongélément ne change pas le sens de la phrase.

Ici, le mot "chapitre" et le numéro de chapitre réel sont de simples passe-partout, et le nom réel du chapitre est marqué par strong :

```
<h1>Chapter 1: <strong>The Praxis</strong></h1>
```

Dans l'exemple suivant, le nom du diagramme dans la légende est balisé avec strong, pour le distinguer du texte passe-partout (avant) et de la description (après) :

```
<figcaption>Figure 1. <strong>Ant colony dynamics</strong>. The ants in this colony are affected by the heat source (upper left) and the food source (lower right).</figcaption>
```

Dans cet exemple, le titre est vraiment "Fleurs, abeilles et miel", mais l'auteur a ajouté un ajout léger au titre. L' strongélément sert ainsi à baliser la première partie pour la distinguer de la dernière partie.

```
<h1><strong>Flowers, Bees, and Honey</strong> and other things I don't understand</h1>
```

Voici un exemple d'avertissement dans un jeu, avec les différentes parties marquées en fonction de leur importance :

```
<p><strong>Warning.</strong> This dungeon is dangerous.
<strong>Avoid the ducks.</strong> Take any gold you find.
<strong><strong>Do not take any of the diamonds</strong>,
they are explosive and <strong>will destroy anything within
ten meters.</strong></strong> You have been warned.</p>
```

Dans cet exemple, l' strongélément est utilisé pour désigner la partie du texte que l'utilisateur est censé lire en premier.

```
<p>Welcome to Remy, the reminder system.</p>
<p>Your tasks for today:</p>
<ul>
```

```
<li><p><strong>Turn off the oven.</strong></p></li>
<li><p>Put out the trash.</p></li>
<li><p>Do the laundry.</p></li>
</ul>
```

#### 4.5.4 L' **small** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML<sup>Element</sup>](#).

L' **small** élément [représente](#) des commentaires annexes tels que des petits caractères.

*Les petits caractères comportent généralement des clauses de non-responsabilité, des mises en garde, des restrictions légales ou des droits d'auteur. Les petits caractères sont également parfois utilisés pour l'attribution ou pour satisfaire aux exigences de licence.*

*L' **small** élément n'atténue pas ou ne diminue pas l'importance du texte mis en valeur par l' **em** élément ou marqué comme important avec l' **strong** élément. Pour marquer le texte comme non souligné ou important, ne le marquez simplement pas avec les éléments **em** ou **strong** respectivement.*

L' **small** élément ne doit pas être utilisé pour des étendues de texte étendues, telles que plusieurs paragraphes, listes ou sections de texte. Il est uniquement destiné aux



courts métrages de texte. Le texte d'une page listant les conditions d'utilisation, par exemple, ne serait pas un candidat approprié pour l' smallélément : dans un tel cas, le texte n'est pas un commentaire annexe, c'est le contenu principal de la page.

L' smallélément ne doit pas être utilisé pour les sous-titres ; pour cela, utilisez l' hgroupélément .

Dans cet exemple, l' smallélément est utilisé pour indiquer que la taxe sur la valeur ajoutée n'est pas incluse dans le prix d'une chambre d'hôtel :

```
<dl>
  <dt>Single room
  <dd>199 € <small>breakfast included, VAT not included</small>
  <dt>Double room
  <dd>239 € <small>breakfast included, VAT not included</small>
</dl>
```

Dans ce deuxième exemple, l' smallélément est utilisé pour un commentaire annexe dans un article.

```
<p>Example Corp today announced record profits for the
second quarter <small>(Full Disclosure: Foo News is a subsidiary of
Example Corp)</small>, leading to speculation about a third quarter
merger with Demo Group.</p>
```

Ceci est distinct d'une barre latérale, qui peut contenir plusieurs paragraphes et est supprimée du flux de texte principal. Dans l'exemple suivant, nous voyons une barre latérale du même article. Cette barre latérale comporte également de petits caractères, indiquant la source des informations dans la barre latérale.

```
<aside>
  <h1>Example Corp</h1>
  <p>This company mostly creates small software and Web
  sites.</p>
  <p>The Example Corp company mission is "To provide entertainment
  and news on a sample basis".</p>
  <p><small>Information obtained from <a
  href="https://example.com/about.html">example.com</a> home
  page.</small></p>
</aside>
```

Dans ce dernier exemple, l' smallélément est marqué comme étant *important* en petits caractères.

```
<p><strong><small>Continued use of this service will result in a
kiss.</small></strong></p>
```

#### 4.5.5 L' **s**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML<sup>Element</sup>](#).

L' **s**élément [représente](#) un contenu qui n'est plus exact ou qui n'est plus pertinent.

*L' **s**élément n'est pas approprié pour indiquer des modifications de document ; pour marquer une portion de texte comme ayant été supprimée d'un document, utilisez l' **del**élément.*

Dans cet exemple, un prix de vente recommandé a été marqué comme n'étant plus pertinent car le produit en question a un nouveau prix de vente.

```
<p>Buy our Iced Tea and Lemonade!</p>  
<p><s>Recommended retail price: $3.99 per bottle</s></p>  
<p><strong>Now selling for just $2.99 a bottle!</strong></p>
```

#### 4.5.6 L' **cite**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .

Contenu palpable .

**Contextes dans lesquels cet élément peut être utilisé :**

Où le contenu du phrasé est attendu.

**Modèle de contenu :**

Contenu de la phrase .

**Omission de balise dans text/html :**

Aucune des deux balises n'est omise.

**Attributs de contenu :**

Attributs globaux

**Considérations d'accessibilité :**

Pour les auteurs .

Pour les exécutants .

**Interface DOM :**

Utilisations HTMLElement.

L' citeélément représente le titre d'une œuvre (par exemple, un livre, un article, un essai, un poème, une partition, une chanson, un scénario, un film, une émission de télévision, un jeu, une sculpture, une peinture, une pièce de théâtre , une pièce de théâtre, un opéra, une comédie musicale, une exposition, un dossier judiciaire, un programme informatique, etc.). Il peut s'agir d'une œuvre citée ou référéncée en détail (c'est-à-dire une citation), ou simplement d'une œuvre mentionnée en passant.

Le nom d'une personne n'est pas le titre d'une œuvre — même si les gens appellent cette personne une œuvre — et l'élément ne doit donc pas être utilisé pour baliser les noms des personnes. (Dans certains cas, l' bélément peut être approprié pour les noms ; par exemple, dans un article de potins où les noms de personnes célèbres sont des mots-clés rendus avec un style différent pour attirer l'attention sur eux. Dans d'autres cas, si un élément est vraiment nécessaire , l' spanélément peut être utilisé.)

L'exemple suivant montre une utilisation typique de l' citeélément :

```
<p>My favorite book is <cite>The Reality Dysfunction</cite> by  
Peter F. Hamilton. My favorite comic is <cite>Pearls Before  
Swine</cite> by Stephan Pastis. My favorite track is <cite>Jive  
Samba</cite> by the Cannonball Adderley Sextet.</p>
```

Voici l'utilisation correcte :

```
<p>According to the Wikipedia article <cite>HTML</cite>, as it  
stood in mid-February 2008, leaving attribute values unquoted is  
unsafe. This is obviously an over-simplification.</p>
```

Ce qui suit, cependant, est un usage incorrect, car l' `<cite>` élément ici contient bien plus que le titre de l'œuvre :

```
<!-- do not copy this example, it is an example of bad usage! -->
<p>According to <cite>the Wikipedia article on HTML</cite>, as it
stood in mid-February 2008, leaving attribute values unquoted is
unsafe. This is obviously an over-simplification.</p>
```

L' `<cite>` élément est un élément clé de toute citation dans une bibliographie, mais il n'est utilisé que pour marquer le titre :

```
<p><cite>Universal Declaration of Human Rights</cite>, United
Nations,
December 1948. Adopted by General Assembly resolution 217 A
(III).</p>
```

*Une citation n'est pas une citation (pour laquelle l' `<q>` élément est approprié).*

C'est une utilisation incorrecte, car `<cite>` n'est pas pour les guillemets :

```
<p><cite>This is wrong!</cite>, said Ian.</p>
```

C'est également un usage incorrect, car une personne n'est pas une œuvre :

```
<p><q>This is still wrong!</q>, said <cite>Ian</cite>.</p>
```

L'utilisation correcte n'utilise pas d' `<cite>` élément :

```
<p><q>This is correct</q>, said Ian.</p>
```

Comme mentionné ci-dessus, l' `<b>` élément peut être pertinent pour marquer des noms comme étant des mots-clés dans certains types de documents :

```
<p>And then <b>Ian</b> said <q>this might be right, in a
gossip column, maybe!</q>.</p>
```

#### 4.5.7 L' `<q>` élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

### **Omission de balise dans text/html** :

Aucune des deux balises n'est omise.

### **Attributs de contenu** :

[Attributs globaux](#)

[cite](#)— Lien vers la source de la citation ou plus d'informations sur la modification

### **Considérations d'accessibilité** :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### **Interface DOM** :

Utilisations [HTMLQuoteElement](#).

L' [q](#)élément [représente](#) un [contenu de phrasé](#) cité d'une autre source.

La ponctuation des guillemets (comme les guillemets) qui cite le contenu de l'élément ne doit pas apparaître immédiatement avant, après ou à l'intérieur [q](#)des éléments ; ils seront insérés dans le rendu par l'agent utilisateur.

Le contenu à l'intérieur d'un [q](#)élément doit être cité d'une autre source, dont l'adresse, si elle en a une, peut être citée dans l' [cite](#)attribut. La source peut être fictive, comme lors de la citation de personnages dans un roman ou un scénario.

Si l' [cite](#)attribut est présent, il doit s'agir d'une [URL valide potentiellement entourée d'espaces](#) . Pour obtenir le lien de citation correspondant, la valeur de l'attribut doit être [analysée](#) par rapport au [nœud document](#) de l'élément . Les agents utilisateurs peuvent autoriser les utilisateurs à suivre ces liens de citation, mais ils sont principalement destinés à un usage privé (par exemple, par des scripts côté serveur collectant des statistiques sur l'utilisation des citations par un site), et non aux lecteurs.

L' [q](#)élément ne doit pas être utilisé à la place des guillemets qui ne représentent pas des guillemets ; par exemple, il est inapproprié d'utiliser l' [q](#)élément pour marquer des déclarations sarcastiques.

L'utilisation d' [q](#)éléments pour baliser les citations est entièrement facultative ; utiliser une ponctuation de citation explicite sans [q](#)éléments est tout aussi correct.

Voici un exemple simple d'utilisation de l' [q](#)élément :

```
<p>The man said <q>Things that are impossible just take  
longer</q>. I disagreed with him.</p>
```

Voici un exemple avec à la fois un lien de citation explicite dans l' [q](#)élément et une citation explicite à l'extérieur :

```
<p>The W3C page <cite>About W3C</cite> says the W3C's mission is <q cite="https://www.w3.org/Consortium/">To lead the World Wide Web to its full potential by developing protocols and guidelines that ensure long-term growth for the Web</q>. I disagree with this mission.</p>
```

Dans l'exemple suivant, la citation elle-même contient une citation :

```
<p>In <cite>Example One</cite>, he writes <q>The man said <q>Things that are impossible just take longer</q>. I disagreed with him</q>. Well, I disagree even more!</p>
```

Dans l'exemple suivant, les guillemets sont utilisés à la place de l' [q](#)élément :

```
<p>His best argument was "I disagree", which I thought was laughable.</p>
```

Dans l'exemple suivant, il n'y a pas de guillemet — les guillemets sont utilisés pour nommer un mot. L'utilisation de l' [q](#)élément dans ce cas serait inappropriée.

```
<p>The word "ineffable" could have been used to describe the disaster resulting from the campaign's mismanagement.</p>
```

#### 4.5.8 L' [dfn](#)élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Phrase content](#) , mais il ne doit y avoir aucun [dfn](#) descendant d'élément.

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
De plus, l' [title](#) attribut [a une sémantique spéciale](#) sur cet élément : Terme complet ou développement d'abréviation.

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

## **Interface DOM :**

Utilisations [HTML](#)[Element](#).

L' [dfn](#)élément [représente](#) l'instance de définition d'un terme. Le [paragraphe](#) , [le groupe de liste de descriptions](#) ou [la section](#) qui est l'ancêtre le plus proche de l' [dfn](#) élément doit également contenir la ou les définitions du [terme](#) donné par l' [dfn](#)élément.

**Terme définissant** : si l' [dfn](#)élément a un [title](#)attribut, alors la valeur exacte de cet attribut est le terme défini. Sinon, s'il contient exactement un nœud enfant d'élément et aucun [Text](#)nœud enfant, et que cet élément enfant est un [abbr](#)élément avec un [title](#)attribut, alors la valeur exacte de cet attribut est le terme défini. Sinon, c'est le [contenu textuel descendant](#) de l' [dfn](#)élément qui donne le terme en cours de définition.

Si l' [title](#)attribut de l' [dfn](#)élément est présent, alors il ne doit contenir que le terme défini.

*L' [title](#)attribut des éléments ancêtres n'affecte pas [dfn](#)les éléments.*

Un [a](#)élément lié à un [dfn](#)élément représente une instance du terme défini par l' [dfn](#)élément.

Dans le fragment suivant, le terme "ouvre-porte de garage" est d'abord défini dans le premier paragraphe, puis utilisé dans le second. Dans les deux cas, c'est son abréviation qui est réellement affichée.

```
<p>The <dfn><abbr title="Garage Door Opener">GDO</abbr></dfn>
is a device that allows off-world teams to open the iris.</p>
<!-- ... later in the document: -->
<p>Teal'c activated his <abbr title="Garage Door Opener">GDO</abbr>
and so Hammond ordered the iris to be opened.</p>
```

Avec l'ajout d'un [a](#)élément, la [référence](#) peut être rendue explicite :

```
<p>The <dfn id=gdo><abbr title="Garage Door
Opener">GDO</abbr></dfn>
is a device that allows off-world teams to open the iris.</p>
<!-- ... later in the document: -->
<p>Teal'c activated his <a href=#gdo><abbr title="Garage Door
Opener">GDO</abbr></a>
and so Hammond ordered the iris to be opened.</p>
```

#### 4.5.9 L' **abbr**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
De plus, l' **title**attribut [a une sémantique spéciale](#) sur cet élément : Terme complet ou développement d'abréviation.

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML\*\*E\*\*lement](#).

L' **abbr**élément [représente](#) une abréviation ou un acronyme, éventuellement avec son expansion. L' **title**attribut peut être utilisé pour fournir une extension de l'abréviation. L'attribut, s'il est spécifié, doit contenir une extension de l'abréviation, et rien d'autre.

Le paragraphe ci-dessous contient une abréviation balisée avec l' **abbr**élément . Ce paragraphe [définit le terme](#) "Groupe de travail sur la technologie des applications hypertextes Web".

```
<p>The <dfn id=whatwg><abbr  
title="Web Hypertext Application Technology Working  
Group">WHATWG</abbr></dfn>  
is a loose unofficial collaboration of web browser manufacturers  
and  
interested parties who wish to develop new technologies designed to  
allow authors to write and deploy Applications over the World Wide  
Web.</p>
```

Une autre façon d'écrire ceci serait:



```
<p>The <dfn id=whatwg>Web Hypertext Application Technology
Working Group</dfn> (<abbr
title="Web Hypertext Application Technology Working
Group">WHATWG</abbr>)
is a loose unofficial collaboration of web browser manufacturers
and
interested parties who wish to develop new technologies designed to
allow authors to write and deploy Applications over the World Wide
Web.</p>
```

Ce paragraphe a deux abréviations. Remarquez comment un seul est défini ; l'autre, sans extension associée, n'utilise pas l' `abbr` élément.

```
<p>The
<abbr title="Web Hypertext Application Technology Working
Group">WHATWG</abbr>
started working on HTML5 in 2004.</p>
```

Ce paragraphe lie une abréviation à sa définition.

```
<p>The <a href="#whatwg"><abbr
title="Web Hypertext Application Technology Working
Group">WHATWG</abbr></a>
community does not have much representation from Asia.</p>
```

Ce paragraphe marque une abréviation sans donner d'expansion, éventuellement comme un crochet pour appliquer des styles pour les abréviations (par exemple, les petites majuscules).

```
<p>Philip` and Dashiva both denied that they were going to
get the issue counts from past revisions of the specification to
backfill the <abbr>WHATWG</abbr> issue graph.</p>
```

Si une abréviation est au pluriel, le numéro grammatical de l'expansion (pluriel vs singulier) doit correspondre au numéro grammatical du contenu de l'élément.

Ici le pluriel est en dehors de l'élément, donc l'expansion est au singulier :

```
<p>Two <abbr title="Working Group">WG</abbr>s worked on
this specification: the <abbr>WHATWG</abbr> and the
<abbr>HTMLWG</abbr>.</p>
```

Ici, le pluriel est à l'intérieur de l'élément, donc l'expansion est au pluriel :

```
<p>Two <abbr title="Working Groups">WGs</abbr> worked on
this specification: the <abbr>WHATWG</abbr> and the
<abbr>HTMLWG</abbr>.</p>
```

Les abréviations n'ont pas besoin d'être marquées à l'aide de cet élément. On s'attend à ce qu'il soit utile dans les cas suivants :

- Abréviations pour lesquelles l'auteur veut donner des extensions, où l'utilisation de l' `abbr` élément avec un `title` attribut est une alternative à l'inclusion de l'extension en ligne (par exemple entre parenthèses).
- Abréviations susceptibles d'être peu familières aux lecteurs du document, pour lesquelles les auteurs sont encouragés soit à baliser l'abréviation à l'aide d'un `abbr` élément avec un `title` attribut, soit à inclure l'expansion en ligne dans le texte la première fois que l'abréviation est utilisée.
- Abréviations dont la présence doit être annotée sémantiquement, par exemple pour être identifiées à partir d'une feuille de style et donner des styles spécifiques, pour lesquels l' `abbr` élément peut être utilisé sans `title` attribut.

Fournir une expansion dans un `title` attribut une seule fois n'entraînera pas nécessairement que les autres `abbr` éléments du même document avec le même contenu mais sans `title` attribut se comportent comme s'ils avaient la même expansion. Chaque `abbr` élément est indépendant.

#### 4.5.10 L' `ruby` élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

Voir prose.

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML`Element`](#).

L' rubyélément permet à une ou plusieurs étendues de contenu de phrasé d'être marquées avec des annotations ruby. Les annotations Ruby sont de courtes séries de texte présentées à côté du texte de base, principalement utilisées dans la typographie d'Asie de l'Est comme guide de prononciation ou pour inclure d'autres annotations. En japonais, cette forme de typographie est également connue sous le nom de *furigana*.

Le modèle de contenu des rubyéléments consiste en une ou plusieurs des séquences suivantes :

1. L'un ou l'autre des éléments suivants :
  - Phrase contenu , mais sans rubyéléments et sans rubydescendants d'éléments
  - Un élément unique rubyqui lui-même n'a aucun rubydescendant d'élément
2. L'un ou l'autre des éléments suivants :
  - Un ou plusieurs rtéléments
  - Un rpélément suivi d'un ou plusieurs rtéléments, dont chacun est lui-même suivi d'un rpélément

Les éléments rubyet rtpeuvent être utilisés pour une variété de types d'annotations, y compris en particulier (mais en aucun cas limité à) celles décrites ci-dessous. Pour plus de détails sur Ruby japonais en particulier, et comment rendre Ruby pour le japonais, consultez *Exigences pour la mise en page du texte japonais* . [\[JLREQ\]](#)

*Au moment de la rédaction, CSS ne fournit pas encore de moyen de contrôler entièrement le rendu de l' rubyélément HTML. Nous espérons que CSS sera étendu pour prendre en charge les styles décrits ci-dessous en temps voulu.*

### **Mono-rubis pour les caractères de base individuels en japonais**

Un ou plusieurs caractères hiragana ou katakana (l'annotation rubis) sont placés avec chaque caractère idéographique (le texte de base). Ceci est utilisé pour fournir des lectures de caractères kanji.

```
<ruby>B<rt>annotation</ruby>
```

Dans cet exemple, notez comment chaque annotation correspond à un seul caractère de base.

```
<ruby>君<rt>くん</ruby><ruby>子<rt>し</ruby>は<ruby>和<rt>わ</ruby>して<ruby>同<rt>どう</ruby>ぜず。
```

くんし わ どう  
君 子は和して 同 ぜず。

Cet exemple peut également être écrit comme suit, en utilisant un élément avec deux segments de texte de base et deux annotations (une pour chacun) plutôt que deux éléments rubydos à dos chacun avec un segment de texte de base et une annotation (comme dans le balisage ci-dessus) ruby:

```
<ruby>君<rt>くん</rt>子<rt>し</rt></ruby>は<ruby>和<rt>わ</rt></ruby>して  
<ruby>同<rt>どう</rt></ruby>ぜず。
```

### Mono-rubis pour les mots composés (jukugo)

Ceci est similaire au cas précédent : chaque caractère idéographique du mot composé (le texte de base) a sa lecture donnée en caractères hiragana ou katakana (l'annotation rubis). La différence est que les segments de texte de base forment un mot composé plutôt que d'être séparés les uns des autres.

```
<ruby>B<rt>annotation</rt>B<rt>annotation</ruby>
```

Dans cet exemple, notez à nouveau comment chaque annotation correspond à un seul caractère de base. Dans cet exemple, chaque mot composé (jukugo) correspond à un seul ruby élément.

Le rendu ici devrait être que chaque annotation soit placée au-dessus (ou à côté, dans le texte vertical) du caractère de base correspondant, les annotations ne surplombant aucun des caractères adjacents.

```
<ruby>鬼<rt>き</rt>門<rt>もん</rt></ruby>の<ruby>方<rt>ほう</rt>  
角<rt>がく</rt></ruby>を<ruby>凝<rt>ぎょう</rt>視<rt>し  
</rt></ruby>する
```

きもん ほうがく ぎょうし  
鬼 門 の 方 角 を 凝 視 する

### Jukugo-rubis

Ceci est sémantiquement identique au cas précédent (chaque caractère idéographique individuel dans le mot composé de base a sa lecture donnée dans une annotation en caractères hiragana ou katakana), mais le rendu est le rendu Jukugo Ruby plus compliqué.

C'est le même exemple que ci-dessus pour mono-ruby pour les mots composés. Le rendu différent devrait être obtenu en utilisant un style différent (par exemple en CSS), et n'est pas illustré ici.

```
<ruby>鬼<rt>き</rt>門<rt>もん</rt></ruby>の<ruby>方<rt>ほう</rt>  
角<rt>がく</rt></ruby>を<ruby>凝<rt>ぎょう</rt>視<rt>し  
</rt></ruby>する
```

Pour plus de détails sur [le rendu Jukugo Ruby](#), voir l'Annexe F dans les Exigences pour la mise en page du texte japonais . [\[JLREQ\]](#)

### Ruby de groupe pour décrire les significations

L'annotation décrit le sens du texte de base, plutôt que (ou en plus) la prononciation. Ainsi, le texte de base et l'annotation peuvent comporter plusieurs caractères.

```
<ruby>BASE<rt>annotation</ruby>
```

Ici, un mot idéographique composé a son katakana correspondant donné sous forme d'annotation.

```
<ruby>境界面<rt>インターフェース</ruby>
```

インターフェース  
境界面

Ici, un mot idéographique composé a sa traduction en anglais fournie sous forme d'annotation.

```
<ruby lang="ja">編集者<rt lang="en">editor</ruby>
```

éditeur  
編集者

### Rubis de groupe pour les lectures de Jukuji

Une lecture phonétique qui correspond à plusieurs caractères de base, car une correspondance un à un serait difficile. (En anglais, les mots "Colonel" et "Lieutenant" sont des exemples de mots où une correspondance directe de la prononciation avec des lettres individuelles est, dans certains dialectes, plutôt floue.)

Dans cet exemple, le nom d'une espèce de fleurs a une lecture phonétique fournie à l'aide du groupe ruby :

```
<ruby>紫陽花<rt>あじさい</ruby>
```

あじさい  
紫陽花

### Texte avec annotations phonétiques et sémantiques (rubis double face)

Parfois, les styles de rubis décrits ci-dessus sont combinés.

Si cela se traduit par deux annotations couvrant le même segment de base unique, les annotations peuvent simplement être placées dos à dos.

```
<ruby>BASE<rt>annotation 1<rt>annotation 2</ruby>
```

```
<ruby>B<rt>a<rt>a</ruby><ruby>A<rt>a<rt>a</ruby><ruby>S<rt>a<rt>a</ruby><ruby>E<rt>a<rt>a</ruby>
```

Dans cet exemple artificiel, certains symboles reçoivent des noms en anglais et en français.

```
<ruby>  
♥ <rt> Heart <rt lang=fr> Cœur </rt>  
♣ <rt> Shamrock <rt lang=fr> Trèfle </rt>  
★ <rt> Star <rt lang=fr> Étoile </rt>  
</ruby>
```

Dans des situations plus compliquées telles que les exemples suivants, un ruby élément imbriqué est utilisé pour donner les annotations internes, puis cet ensemble ruby reçoit une annotation au niveau "externe".

```
<ruby><ruby>B<rt>a</rt>A<rt>n</rt>S<rt>t</rt>E<rt>n</rt></ruby><rt>annotation</ruby>
```

Ici, la lecture phonétique et la signification sont données en annotations rubis. L'annotation sur l' rubyélément imbriqué donne une annotation phonétique mono-rubis pour chaque caractère de base, tandis que l'annotation dans l' rtélément qui est un enfant de l' rubyélément externe donne la signification en utilisant les hiragana.

```
<ruby><ruby>東<rt>とう</rt>南<rt>なん</rt></ruby><rt>たつみ</rt></ruby>の方角
```

とうなん  
東 南 たつみの方角

C'est le même exemple, mais la signification est donnée en anglais au lieu du japonais :

```
<ruby><ruby>東<rt>とう</rt>南<rt>なん</rt></ruby><rt lang=en>Southeast</rt></ruby>の方角
```

とうなん  
東 南 Sud-estの方角

---

Dans un rubyélément qui n'a pas d' rubyancêtre d'élément, le contenu est segmenté et les segments sont placés dans trois catégories : segments de texte de base, segments d'annotation et segments ignorés. Les segments ignorés ne font pas partie de la sémantique du document (ils consistent en des espaces inter-éléments et rp , ces derniers étant utilisés pour les agents utilisateurs hérités qui ne prennent pas du tout en charge ruby). Les segments de texte de base peuvent se chevaucher (avec une limite de deux segments chevauchant n'importe quelle position dans le DOM, et avec tout segment ayant un point de départ antérieur à un segment qui se chevauche ayant également un point final égal ou ultérieur, et tout segment ayant un point final ultérieur qu'un segment superposé ayant également un point de départ égal ou antérieur). Les segments d'annotation correspondent à rt éléments. Chaque segment d'annotation peut être associé à un segment de texte de base, et chaque segment de texte de base peut être associé à des segments d'annotation. (Dans un document conforme, chaque segment de texte de base est associé à au moins un segment d'annotation, et chaque segment d'annotation est associé à un segment de texte de base.) A ruby représente l' union des segments de texte de base qu'il contient, ainsi que le mappage de ces segments de texte de base aux segments d'annotation. Les segments sont décrits en termes de plages DOM ; les plages de segments d'annotation se composent toujours d'exactly un élément. [\[DOM\]](#)

À tout moment, la segmentation et la catégorisation du contenu d'un ruby élément est le résultat qui serait obtenu en exécutant l'algorithme suivant :

1. Soit *les segments de texte de base* une liste vide de segments de texte de base, chacun potentiellement avec une liste de sous-segments de texte de base.
2. Soit *les segments d'annotation* une liste vide de segments d'annotation, chacun étant potentiellement associé à un segment ou sous-segment de texte de base.
3. Soit *root* l' rubyélément pour lequel l'algorithme est exécuté.
4. Si *root* a un rubyancêtre d'élément, passez à l'étape intitulée *end* .
5. Soit *le parent actuel* *root* .
6. Soit *indice* 0.
7. Laissez *l'index de début* être nul.
8. Laissez *l'index de début du parent* être nul.
9. Laissez *le texte de base actuel* être nul.
10. *Mode de démarrage* : si *l'index* est égal ou supérieur au nombre de nœuds enfants dans *le parent actuel* , passez à l'étape intitulée *mode de fin* .
11. Si l' *index* e nœud dans *le parent actuel* est un élément rt ou rp, passez à l'étape intitulée *mode d'annotation* .
12. Définissez *l'index de départ* sur la valeur de *index* .
13. *Mode de base* : Si l' *index* e nœud dans *le parent actuel* est un rubyélément, et si *le parent actuel* est le même élément que *root* , alors poussez un niveau ruby puis sautez à l'étape étiquetée *start mode* .
14. Si l' *index* e nœud dans *le parent actuel* est un élément rt ou , définissez le texte de base actuel , puis passez à l'étape intitulée *mode d'annotation* .rp
15. Incrémenter *l'index* de un.
16. *Post-incrémentation du mode de base* : si *index* est égal ou supérieur au nombre de nœuds enfants dans *le parent actuel* , passez à l'étape intitulée *end mode* .
17. Revenez à l'étape intitulée *mode de base* .
18. *Mode d'annotation* : si l' *index* e nœud dans *le parent actuel* est un rtélément, alors poussez une annotation ruby et passez à l'étape intitulée *mode d'annotation incrément* .
19. Si l' *index* e nœud dans *le parent actuel* est un rpélément, passez à l'étape étiquetée *incrémentation du mode d'annotation* .

20. If the *index*th node in *current parent* is not a Text node, or is a Text node that is not inter-element whitespace, then jump to the step labeled *base mode*.
21. *Annotation mode increment*: Let *lookahead index* be *index* plus one.
22. *Annotation mode white-space skipper*: If *lookahead index* is equal to the number of child nodes in *current parent* then jump to the step labeled *end mode*.
23. Si l' *index d'anticipation* e nœud dans le *parent actuel* est un rtélément ou un rpélément, définissez l'*index* sur l'*index d'anticipation* et passez à l'étape étiquetée *mode d'annotation* .
24. Si l' *index d'anticipation* e nœud dans le *parent actuel* n'est pas un Textnœud, ou est un Textnœud qui n'est pas inter-élément whitespace , alors passez à l'étape étiquetée *mode de base* (sans incrémenter davantage *index* , donc l' inter-élément whitespace vu jusqu'à présent devient partie du segment de texte de base suivant).
25. Incrémente l'*index d'anticipation* d'une unité.
26. Passez à l'étape intitulée *annotation mode white-space skipper* .
27. *Mode de fin* : Si le *parent actuel* n'est pas le même élément que *root* , alors sautez un niveau ruby et sautez à l'étape étiquetée *base mode post-increment* .
28. *Fin* : Renvoie les segments de texte de base et les segments d'*annotation* . Tout contenu de l' rubyélément non décrit par des segments dans l'une de ces listes est implicitement dans un *segment ignoré* .

Lorsque les étapes ci-dessus indiquent de **définir le texte de base actuel** , cela signifie exécuter les étapes suivantes à ce stade de l'algorithme :

1. Soit *la plage de texte* une plage DOM dont le début est le point limite ( *parent actuel* , *index de début* ) et dont la fin est le point limite ( *parent actuel* , *index* ).
2. Soit *nouveau segment de texte* un segment de texte de base décrit par l' *annotation de plage range* .
3. Ajouter un nouveau segment de texte aux segments de texte de base .
4. Soit le *texte de base actuel* comme *nouveau segment de texte* .
5. Laissez l'*index de début* être nul.

Lorsque les étapes ci-dessus indiquent de **pousser un ruby level** , cela signifie exécuter les étapes suivantes à ce stade de l'algorithme :

1. Soit *parent actuel* l' *index* e nœud dans *parent actuel* .



2. Soit *indice* 0.
3. Définissez *l'index de démarrage enregistré* sur la valeur de *l'index de démarrage* .
4. Laissez *l'index de début* être nul.

Lorsque les étapes ci-dessus indiquent de **faire apparaître un niveau ruby** , cela signifie exécuter les étapes suivantes à ce stade de l'algorithme :

1. Soit *index* la position du *parent actuel* dans *root* .
2. Soit *le parent actuel* *root* .
3. Incrémenter *l'index* de un.
4. Définissez *l'index de démarrage* sur la valeur de *l'index de démarrage enregistré* .
5. Laissez *l'index de début enregistré* être nul.

Lorsque les étapes ci-dessus indiquent de **pousser une annotation ruby** , cela signifie exécuter les étapes suivantes à ce stade de l'algorithme :

1. Soit *rt* l' rtélément qui est l' *index* ème nœud du *parent courant* .
2. Soit *la plage d'annotations* une plage DOM dont le début est le point limite ( *parent actuel* , *index* ) et dont la fin est le point limite ( *parent actuel* , *index* plus un ) (c'est-à-dire qui ne contient que *rt* ).
3. Soit *new annotation segment* un segment d'annotation décrit par la *plage annotation range* .
4. Si *le texte de base actuel* n'est pas nul, associez *le nouveau segment d'annotation* au *texte de base actuel* .
5. Ajouter un nouveau segment d'annotation aux segments d'annotation .

Dans cet exemple, chaque idéogramme du texte japonais 漢字 est annoté avec sa lecture en hiragana.

```
...
<ruby>漢<rt>かん</rt>字<rt>じ</rt></ruby>
...
```

Cela pourrait être rendu comme suit :

かん じ  
... 漢字 ...

Dans cet exemple, chaque idéogramme du texte chinois traditionnel 漢字 est annoté avec sa lecture bopomofo.

```
<ruby>漢<rt>ㄏㄢˋ</rt>字<rt>ㄗㄩˋ</rt></ruby>
```

Cela pourrait être rendu comme suit :

漢 ㄏㄢˋ  
字 ㄗㄩˋ

Dans cet exemple, chaque idéogramme du texte chinois simplifié 汉字 est annoté avec sa lecture en pinyin.

```
...<ruby>汉<rt>hàn</rt>字<rt>zì</rt></ruby>...
```

Cela pourrait être rendu comme suit :

hàn zì  
... 汉字 ...

Dans cet exemple plus artificiel, l'acronyme "HTML" a quatre annotations : une pour l'ensemble de l'acronyme, décrivant brièvement ce qu'il est, une pour les lettres "HT" en les développant en "Hypertexte", une pour la lettre "M" en les développant à "Markup", et une pour la lettre "L" en l'étendant à "Langage".

```
<ruby>  
  
<ruby>HT<rt>Hypertext</rt>M<rt>Markup</rt>L<rt>Language</rt></ruby>  
<rt>An abstract language for describing documents and applications  
</ruby>
```

#### 4.5.11 L' **rt** élément

✓ MDN

Catégories :

Aucun.

**Contextes dans lesquels cet élément peut être utilisé :**

En tant qu'enfant d'un [ruby](#)élément.

**Modèle de contenu :**

[Contenu de la phrase](#) .

**Omission de balise dans text/html :**

[La balise de fin](#) d'un [rt](#)élément peut être omise si l'élément est immédiatement suivi d'un élément ou , ou s'il n'y a plus de contenu dans l'élément parent.[rt](#)[rt](#)[rp](#)

**Attributs de contenu :**

[Attributs globaux](#)

**Considérations d'accessibilité :**

[Pour les auteurs](#) .

[Pour les exécutants](#) .

**Interface DOM :**

Utilisations [HTML](#)[Element](#).

L' [rt](#)élément marque le composant de texte ruby d'une annotation ruby. Lorsqu'il est l'enfant d'un [ruby](#)élément, il ne [représente](#) rien lui-même, mais l' [ruby](#)élément l'utilise pour déterminer ce *qu'il* [représente](#) .

Un [rt](#)élément qui n'est pas un enfant d'un [ruby](#)élément [représente](#) la même chose que ses enfants.

#### 4.5.12 L' [rp](#)élément



**Catégories :**

Aucun.

**Contextes dans lesquels cet élément peut être utilisé :**

En tant qu'enfant d'un [ruby](#)élément, soit immédiatement avant, soit immédiatement après un [rt](#)élément.

**Modèle de contenu :**

[Texte](#) .

**Omission de balise dans text/html :**

[La balise de fin](#) d'un [rp](#)élément peut être omise si l'élément est immédiatement suivi d'un élément ou , ou s'il n'y a plus de contenu dans l'élément parent.[rp](#)[rt](#)[rp](#)

### [Attributs de contenu](#) :

[Attributs globaux](#)

### [Considérations d'accessibilité](#) :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### [Interface DOM](#) :

Utilisations [HTML](#)[Element](#).

L' [rp](#)élément peut être utilisé pour fournir des parenthèses ou un autre contenu autour d'un composant de texte ruby d'une annotation ruby, à afficher par les agents utilisateurs qui ne prennent pas en charge les annotations ruby.

Un [rp](#)élément enfant d'un [ruby](#) élément ne [représente](#) rien. Un [rp](#)élément dont l'élément parent n'est pas un [ruby](#)élément [représente](#) ses enfants.

L'exemple ci-dessus, dans lequel chaque idéogramme du texte 漢字 est annoté avec sa lecture phonétique, pourrait être étendu pour être utilisé [rp](#) de sorte que dans les anciens agents utilisateurs, les lectures soient entre parenthèses :

```
...
<ruby>漢<rp> ( </rp><rt>かん</rt><rp> ) </rp>字<rp> ( </rp><rt>じ
</rt><rp> ) </rp></ruby>
...
```

Dans les agents utilisateurs conformes, le rendu serait comme ci-dessus, mais dans les agents utilisateurs qui ne prennent pas en charge ruby, le rendu serait :

... 漢 (かん) 字 (じ) ...

Lorsqu'il existe plusieurs annotations pour un segment, [rp](#)des éléments peuvent également être placés entre les annotations. Voici une autre copie d'un exemple antérieur montrant des symboles avec des noms donnés en anglais et en français, mais cette fois avec [rp](#)des éléments également :

```
<ruby>
♥<rp>: </rp><rt>Heart</rt><rp>, </rp><rt>
lang=fr>Cœur</rt><rp>.</rp>
♣<rp>: </rp><rt>Shamrock</rt><rp>, </rp><rt>
lang=fr>Trèfle</rt><rp>.</rp>
★<rp>: </rp><rt>Star</rt><rp>, </rp><rt>
lang=fr>Étoile</rt><rp>.</rp>
</ruby>
```

Cela rendrait l'exemple rendu comme suit dans les agents utilisateurs non compatibles avec Ruby :

♥ : Coeur, Coeur . ♣ : Shamrock, Trèfle . ★ : Étoile, Étoile .

#### 4.5.13 L' **data**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
[value](#)— Valeur lisible par machine

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLDataElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute DOMString value;
```

```
};
```

L' dataélément représente son contenu, ainsi qu'une forme lisible par machine de ce contenu dans l' value attribut.

L' valueattribut doit être présent. Sa valeur doit être une représentation du contenu de l'élément dans un format lisible par machine.

*Lorsque la valeur est liée à la date ou à l'heure, l' time élément le plus spécifique peut être utilisé à la place.*

L'élément peut être utilisé à plusieurs fins.

Lorsqu'il est combiné avec des microformats ou les attributs de microdonnées définis dans cette spécification, l'élément sert à fournir à la fois une valeur lisible par machine aux fins des processeurs de données et une valeur lisible par l'homme aux fins de rendu dans un navigateur Web. Dans ce cas, le format à utiliser dans l' valueattribut est déterminé par les microformats ou le vocabulaire de microdonnées utilisé.

Cependant, l'élément peut également être utilisé conjointement avec des scripts dans la page, lorsqu'un script a une valeur littérale à stocker à côté d'une valeur lisible par l'homme. Dans de tels cas, le format à utiliser dépend uniquement des besoins du script. (Les data-\* attributs peuvent également être utiles dans de telles situations.)



L' valueattribut IDL doit refléter l'attribut de contenu du même nom.

Ici, un tableau court a ses valeurs numériques codées à l'aide de l' dataélément afin que la bibliothèque JavaScript de tri de tableau puisse fournir un mécanisme de tri sur chaque colonne malgré que les nombres soient présentés sous forme textuelle dans une colonne et sous une forme décomposée dans une autre.

```
<script src="sortable.js"></script>
<table class="sortable">
  <thead> <tr> <th> Game <th> Corporations <th> Map Size
  <tbody>
    <tr> <td> 1830 <td> <data value="8">Eight</data> <td> <data
value="93">19+74 hexes (93 total)</data>
    <tr> <td> 1856 <td> <data value="11">Eleven</data> <td> <data
value="99">12+87 hexes (99 total)</data>
    <tr> <td> 1870 <td> <data value="10">Ten</data> <td> <data
value="149">4+145 hexes (149 total)</data>
  </table>
```

#### 4.5.14 L' **time** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

Si l'élément a un **datetime** attribut : [Phrasing content](#) .  
Sinon : [Texte](#) , mais doit correspondre aux exigences décrites dans le texte ci-dessous.

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
**datetime** — Valeur lisible par machine

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLTimeElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute DOMString dateTime;
```

```
};
```

L' **time** élément [représente](#) son contenu, ainsi qu'une forme lisible par machine de ce contenu dans l' **datetime** attribut. Le type de contenu est limité à différents types de dates, d'heures, de décalages de fuseaux horaires et de durées, comme décrit ci-dessous.

L' **datetime** attribut peut être présent. S'il est présent, sa valeur doit être une représentation du contenu de l'élément dans un format lisible par machine.

Un [time](#)élément qui n'a pas d' [datetime](#)attribut content ne doit avoir aucun descendant d'élément.

La **valeur datetime** d'un [time](#)élément est la valeur de l' [datetime](#)attribut content de l'élément, s'il en a un, sinon le [contenu du texte enfant](#) de l' [time](#)élément.

La [valeur datetime](#) d'un [time](#)élément doit correspondre à l'une des syntaxes suivantes.

#### Une [chaîne de mois valide](#)

```
<time>2011-11</time>
```

#### Une [chaîne de date valide](#)

```
<time>2011-11-18</time>
```

#### Une [chaîne de date valide sans année](#)

```
<time>11-18</time>
```

#### Une [chaîne de temps valide](#)

```
<time>14:54</time>  
<time>14:54:39</time>  
<time>14:54:39.929</time>
```

#### Une [chaîne de date et d'heure locale valide](#)

```
<time>2011-11-18T14:54</time>  
<time>2011-11-18T14:54:39</time>  
<time>2011-11-18T14:54:39.929</time>  
<time>2011-11-18 14:54</time>  
<time>2011-11-18 14:54:39</time>  
<time>2011-11-18 14:54:39.929</time>
```

*Les heures avec des dates mais sans décalage de fuseau horaire sont utiles pour spécifier des événements qui sont observés à la même heure spécifique dans chaque fuseau horaire, tout au long d'une journée. Par exemple, le nouvel an 2020 est célébré au 2020-01-01 00:00 dans chaque fuseau horaire, et non au même moment précis dans tous les fuseaux horaires. Pour les événements qui se produisent en même temps dans tous les fuseaux horaires, par exemple une réunion de vidéoconférence, une [chaîne de date et d'heure globale valide](#) est probablement plus utile.*

#### Une [chaîne de décalage de fuseau horaire valide](#)

```
<time>Z</time>  
<time>+0000</time>  
<time>+00:00</time>  
<time>-0800</time>  
<time>-08:00</time>
```

*Pour les heures sans dates (ou les heures faisant référence à des événements qui se reproduisent à plusieurs dates), il est généralement plus utile de spécifier l'emplacement géographique qui contrôle l'heure que de spécifier un décalage de fuseau horaire, car les emplacements géographiques modifient les décalages de fuseau horaire avec l'heure d'été. Dans certains cas, les emplacements géographiques changent même de fuseau horaire, par exemple lorsque les limites de ces fuseaux horaires sont redessinées, comme cela s'est produit avec Samoa à la fin de 2011. Il existe une base de données*



de fuseaux horaires qui décrit les limites des fuseaux horaires et les règles qui s'y appliquent. chacune de ces zones, connue sous le nom de base de données de fuseaux horaires . [\[ZDATABASE\]](#)

### Une chaîne de date et d'heure globale valide

```
<time>2011-11-18T14:54Z</time>
<time>2011-11-18T14:54:39Z</time>
<time>2011-11-18T14:54:39.929Z</time>
<time>2011-11-18T14:54+0000</time>
<time>2011-11-18T14:54:39+0000</time>
<time>2011-11-18T14:54:39.929+0000</time>
<time>2011-11-18T14:54+00:00</time>
<time>2011-11-18T14:54:39+00:00</time>
<time>2011-11-18T14:54:39.929+00:00</time>
<time>2011-11-18T06:54-0800</time>
<time>2011-11-18T06:54:39-0800</time>
<time>2011-11-18T06:54:39.929-0800</time>
<time>2011-11-18T06:54-08:00</time>
<time>2011-11-18T06:54:39-08:00</time>
<time>2011-11-18T06:54:39.929-08:00</time>
<time>2011-11-18 14:54Z</time>
<time>2011-11-18 14:54:39Z</time>
<time>2011-11-18 14:54:39.929Z</time>
<time>2011-11-18 14:54+0000</time>
<time>2011-11-18 14:54:39+0000</time>
<time>2011-11-18 14:54:39.929+0000</time>
<time>2011-11-18 14:54+00:00</time>
<time>2011-11-18 14:54:39+00:00</time>
<time>2011-11-18 14:54:39.929+00:00</time>
<time>2011-11-18 06:54-0800</time>
<time>2011-11-18 06:54:39-0800</time>
<time>2011-11-18 06:54:39.929-0800</time>
<time>2011-11-18 06:54-08:00</time>
<time>2011-11-18 06:54:39-08:00</time>
<time>2011-11-18 06:54:39.929-08:00</time>
```

Les heures avec des dates et un décalage de fuseau horaire sont utiles pour spécifier des événements spécifiques ou des événements virtuels récurrents où l'heure n'est pas ancrée à un emplacement géographique spécifique. Par exemple, l'heure précise d'un impact d'astéroïde, ou une réunion particulière dans une série de réunions tenues à 1400 UTC tous les jours, qu'une partie particulière du monde observe ou non l'heure d'été. Pour les événements où l'heure précise varie en fonction du décalage du fuseau horaire local d'un emplacement géographique spécifique, une [chaîne de date et d'heure locale valide](#) combinée à cet emplacement géographique est probablement plus utile.

### Une chaîne de semaine valide

```
<time>2011-W47</time>
```

### Quatre chiffres ASCII ou plus , dont au moins un n'est pas U+0030 CHIFFRE ZÉRO (0)

```
<time>2011</time>
<time>0001</time>
```

### Une chaîne de durée valide

```
<time>PT4H18M3S</time>
<time>4h 18m 3s</time>
```

L' **équivalent lisible par machine du contenu de l'élément** doit être obtenu à partir de la [valeur datetime](#) de l'élément en utilisant l'algorithme suivant :

1. Si [l'analyse d'une chaîne de mois à partir de la valeur datetime](#) de l'élément renvoie un [month](#) , c'est l'équivalent lisible par machine ; retour.
2. Si [l'analyse d'une chaîne de date à partir de la valeur datetime](#) de l'élément renvoie une [date](#) , c'est l'équivalent lisible par machine ; retour.
3. Si [l'analyse d'une chaîne de date sans année à partir de la valeur datetime](#) de l'élément renvoie une [date sans année](#) , c'est l'équivalent lisible par machine ; retour.
4. Si [l'analyse d'une chaîne d'heure à partir de la valeur datetime](#) de l'élément renvoie un [time](#) , c'est l'équivalent lisible par machine ; retour.
5. Si [l'analyse d'une chaîne de date et d'heure locale à partir de la valeur datetime](#) de l'élément renvoie une [date et une heure locales](#) , c'est l'équivalent lisible par machine ; retour.
6. Si [l'analyse d'une chaîne de décalage de fuseau horaire à partir de la valeur datetime](#) de l'élément renvoie un [décalage de fuseau horaire](#) , c'est l'équivalent lisible par machine ; retour.
7. Si [l'analyse d'une chaîne de date et d'heure globale à partir de la valeur datetime](#) de l'élément renvoie une [date et une heure globales](#) , il s'agit de l'équivalent lisible par machine ; retour.
8. Si [l'analyse d'une chaîne de semaine à partir de la valeur datetime](#) de l'élément renvoie une [semaine](#) , c'est l'équivalent lisible par machine ; retour.
9. Si la [valeur datetime](#) de l'élément se compose uniquement de [chiffres ASCII](#) , dont au moins un n'est pas U+0030 DIGIT ZERO (0), alors l'équivalent lisible par machine est l'interprétation en base dix de ces chiffres, représentant une année ; retour.
10. Si [l'analyse d'une chaîne de durée à partir de la valeur datetime](#) de l'élément renvoie une [duration](#) , c'est l'équivalent lisible par machine ; retour.
11. Il n'y a pas d'équivalent lisible par machine.

*Les algorithmes référencés ci-dessus sont destinés à être conçus de telle sorte que pour toute chaîne arbitraire s , un seul des algorithmes renvoie une valeur. Une approche plus efficace pourrait consister à créer un algorithme unique qui analyse tous ces types de données en une seule passe ; le développement d'un tel algorithme est laissé en exercice au lecteur.*



L' **dateTime** attribut IDL doit [refléter](#) l'attribut de contenu de l'élément [datetime](#).

L' timeélément peut être utilisé pour encoder des dates, par exemple dans des microformats. Ce qui suit montre une manière hypothétique d'encoder un événement en utilisant une variante sur hCalendar qui utilise l' timeélément :

```
<div class="vevent">
  <a class="url"
href="http://www.web2con.com/">http://www.web2con.com/</a>
  <span class="summary">Web 2.0 Conference</span>:
  <time class="dtstart" datetime="2005-10-05">October 5</time> -
  <time class="dtend" datetime="2005-10-07">7</time>,
  at the <span class="location">Argent Hotel, San Francisco,
CA</span>
</div>
```

Ici, un vocabulaire de microdonnées fictif basé sur le vocabulaire Atom est utilisé avec l' timeélément pour marquer la date de publication d'un article de blog.

```
<article itemscope itemtype="https://n.example.org/rfc4287">
  <h1 itemprop="title">Big tasks</h1>
  <footer>Published <time itemprop="published" datetime="2009-08-
29">two days ago</time>.</footer>
  <p itemprop="content">Today, I went out and bought a bike for my
kid.</p>
</article>
```

Dans cet exemple, la date de publication d'un autre article est balisée à l'aide de time, cette fois en utilisant le vocabulaire des microdonnées schema.org :

```
<article itemscope itemtype="http://schema.org/BlogPosting">
  <h1 itemprop="headline">Small tasks</h1>
  <footer>Published <time itemprop="datePublished" datetime="2009-
08-30">yesterday</time>.</footer>
  <p itemprop="articleBody">I put a bike bell on her bike.</p>
</article>
```

Dans l'extrait de code suivant, l' timeélément est utilisé pour encoder une date au format ISO8601, pour un traitement ultérieur par un script :

```
<p>Our first date was <time datetime="2006-09-23">a
Saturday</time>.</p>
```

Dans ce deuxième extrait, la valeur inclut une heure :

```
<p>We stopped talking at <time datetime="2006-09-24T05:00-
07:00">5am the next morning</time>.</p>
```

Un script chargé par la page (et donc au courant de la convention interne de la page de marquer les dates et les heures à l'aide de l' timeélément) pourrait parcourir la

page et regarder tous les [time](#) éléments qu'elle contient pour créer un index des dates et des heures.

Par exemple, cet élément transmet la chaîne "vendredi" avec la sémantique supplémentaire que le 18 novembre 2011 est la signification qui correspond à "vendredi" :

```
Today is <time datetime="2011-11-18">Friday</time>.
```

Dans cet exemple, une heure spécifique dans le fuseau horaire de l'heure normale du Pacifique est spécifiée :

```
Your next meeting is at <time datetime="2011-11-18T15:00-08:00">3pm</time>.
```

#### 4.5.15 L' [code](#) élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML](#) [Element](#).

L' [code](#) élément [représente](#) un fragment de code informatique. Il peut s'agir d'un nom d'élément XML, d'un nom de fichier, d'un programme informatique ou de toute autre chaîne reconnue par un ordinateur.

There is no formal way to indicate the language of computer code being marked up. Authors who wish to mark [code](#) elements with the language used, e.g. so that syntax

highlighting scripts can use the right rules, can use the [class](#) attribute, e.g. by adding a class prefixed with "language-" to the element.

The following example shows how the element can be used in a paragraph to mark up element names and computer code, including punctuation.

```
<p>The <code>code</code> element represents a fragment of computer code.</p>

<p>When you call the <code>activate()</code> method on the <code>robotSnowman</code> object, the eyes glow.</p>

<p>The example below uses the <code>begin</code> keyword to indicate the start of a statement block. It is paired with an <code>end</code> keyword, which is followed by the <code>.</code> punctuation character (full stop) to indicate the end of the program.</p>
```

The following example shows how a block of code could be marked up using the [pre](#) and [code](#) elements.

```
<pre><code class="language-pascal">var i: Integer;
begin
  i := 1;
end.</code></pre>
```

A class is used in that example to indicate the language used.

*See the [pre](#) element for more details.*

#### 4.5.16 The **var** element



##### **Categories:**

[Flow content.](#)  
[Phrasing content.](#)  
[Palpable content.](#)

##### **Contexts in which this element can be used:**

Where [phrasing content](#) is expected.

##### **Content model:**

[Phrasing content.](#)

### Tag omission in text/html:

Neither tag is omissible.

### Content attributes:

Global attributes

### Accessibility considerations:

For authors.

For implementers.

### DOM interface:

Uses HTML~~E~~lement.

L' varélément représente une variable. Il peut s'agir d'une variable réelle dans une expression mathématique ou dans un contexte de programmation, d'un identifiant représentant une constante, d'un symbole identifiant une quantité physique, d'un paramètre de fonction ou simplement d'un terme utilisé comme espace réservé en prose.

Dans le paragraphe ci-dessous, la lettre "n" est utilisée comme variable en prose :

```
<p>If there are <var>n</var> pipes leading to the ice  
cream factory then I expect at <em>least</em> <var>n</var>  
flavors of ice cream to be available for purchase!</p>
```

Pour les mathématiques, en particulier pour tout ce qui va au-delà des expressions les plus simples, MathML est plus approprié. Cependant, l' varélément peut toujours être utilisé pour faire référence à des variables spécifiques qui sont ensuite mentionnées dans les expressions MathML.

Dans cet exemple, une équation est affichée, avec une légende qui fait référence aux variables de l'équation. L'expression elle-même est balisée avec MathML, mais les variables sont mentionnées dans la légende de la figure à l'aide de var.

```
<figure>  
  <math>  
    <mi>a</mi>  
    <mo>=</mo>  
    <msqrt>  
      <msup><mi>b</mi><mn>2</mn></msup>  
      <mi>+</mi>  
      <msup><mi>c</mi><mn>2</mn></msup>  
    </msqrt>  
  </math>  
  <figcaption>  
    Using Pythagoras' theorem to solve for the hypotenuse  
<var>a</var> of
```

```
a triangle with sides <var>b</var> and <var>c</var>
</figcaption>
</figure>
```

Ici, l'équation décrivant l'équivalence masse-énergie est utilisée dans une phrase, et l' `var` élément est utilisé pour marquer les variables et les constantes dans cette équation :

```
<p>Then she turned to the blackboard and picked up the chalk. After
a few moment's
thought, she wrote <var>E</var> = <var>m</var>
<var>c</var><sup>2</sup>. The teacher
looked pleased.</p>
```

#### 4.5.17 L' `samp` élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML`Element`](#).

L' `samp` élément [représente](#) un exemple ou une sortie citée d'un autre programme ou système informatique.

*Voir les éléments [pre](#) et [kbd](#) pour plus de détails.  
Cet élément peut être mis en contraste avec l' [output](#) élément, qui peut être utilisé pour fournir une sortie immédiate dans une application Web.*

Cet exemple montre l' [samp](#)élément utilisé en ligne :

```
<p>The computer said <samp>Too much cheese in tray  
two</samp> but I didn't know what that meant.</p>
```

Ce deuxième exemple montre un bloc d'échantillons de sortie d'un programme de console. [samp](#)Les éléments et imbriqués [kbd](#)permettent de styliser des éléments spécifiques de l'exemple de sortie à l'aide d'une feuille de style. Il y a aussi quelques parties du [samp](#)qui sont annotées avec un balisage encore plus détaillé, pour permettre un style très précis. Pour y parvenir, [span](#)des éléments sont utilisés.

```
<pre><samp><span class="prompt">jdoe@mowmow:~$</span> <kbd>ssh  
demo.example.com</kbd>  
  
Last login: Tue Apr 12 09:10:17 2005 from mowmow.example.com on  
pts/1  
  
Linux demo 2.6.10-grsec+gg3+e+fhs6b+nfs+gr0501+++p3+c4a+gr2b-  
reslog-v6.189 #1 SMP Tue Feb 1 11:22:36 PST 2005 i686 unknown  
  
<span class="prompt">jdoe@demo:~$</span> <span  
class="cursor">_</span></samp></pre>
```

Ce troisième exemple montre un bloc d'entrée et sa sortie respective. L'exemple utilise à la fois les éléments [code](#)et [samp](#).

```
<pre>  
<code class="language-javascript">console.log(2.3 + 2.4)</code>  
<samp>4.699999999999999</samp>  
</pre>
```

#### 4.5.18 L' [kbd](#)élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.



## Attributs de contenu :

Attributs globaux

## Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

## Interface DOM :

Utilisations HTML Element.

L' kbdélément représente l'entrée de l'utilisateur (généralement une entrée au clavier, bien qu'il puisse également être utilisé pour représenter d'autres entrées, telles que des commandes vocales).

Lorsque l' kbdélément est imbriqué dans un sampélément, il représente l'entrée telle qu'elle a été renvoyée par le système.

Lorsque l' kbdélément *contient* un sampélément, il représente une entrée basée sur la sortie du système, par exemple en invoquant un élément de menu.

Lorsque l' kbdélément est imbriqué dans un autre kbdélément, il représente une clé réelle ou une autre unité unique d'entrée selon le mécanisme d'entrée.

Ici, l' kbdélément est utilisé pour indiquer les touches à appuyer :

```
<p>To make George eat an apple, press <kbd><kbd>Shift</kbd> +  
<kbd>F3</kbd></kbd></p>
```

Dans ce deuxième exemple, l'utilisateur est invité à choisir un élément de menu particulier. L' kbdélément externe marque un bloc d'entrée, les kbdéléments internes représentant chaque étape individuelle de l'entrée, et les sampéléments à l'intérieur indiquant que les étapes sont entrées en fonction de quelque chose affiché par le système, dans ce cas les étiquettes de menu :

```
<p>To make George eat an apple, select  
  <kbd><kbd><samp>File</samp></kbd>|<kbd><samp>Eat  
Apple...</samp></kbd></kbd>  
</p>
```

Une telle précision n'est pas nécessaire ; ce qui suit est tout aussi bien :

```
<p>To make George eat an apple, select <kbd>File | Eat  
Apple...</kbd></p>
```

### 4.5.19 Les éléments subetsup



## Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

## Modèle de contenu :

[Contenu de la phrase](#) .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

[Attributs globaux](#)

## Considérations d'accessibilité :

L' [sub](#)élément : [pour les auteurs](#) ; [pour les exécutants](#) .  
L' [sup](#)élément : [pour les auteurs](#) ; [pour les exécutants](#) .

## Interface DOM :

Utilisez [HTML<sup>Element</sup>](#).

L' [sup](#)élément [représente](#) un exposant et l' [sub](#) élément [représente](#) un indice.

Ces éléments doivent être utilisés uniquement pour marquer les conventions typographiques avec des significations spécifiques, et non pour la présentation typographique pour le plaisir de la présentation. Par exemple, il serait inapproprié que les éléments [sub](#) et [sup](#) soient utilisés dans le nom du système de préparation de documents LaTeX. En général, les auteurs ne doivent utiliser ces éléments que si l' *absence* de ces éléments modifie le sens du contenu.

Dans certaines langues, les exposants font partie des conventions typographiques pour certaines abréviations.

```
<p>Their names are  
<span lang="fr"><abbr>M<sup>lle</sup></abbr> Gwendoline</span> and  
<span lang="fr"><abbr>M<sup>me</sup></abbr> Denise</span>.</p>
```

L' [sub](#)élément peut être utilisé à l'intérieur d'un [var](#)élément, pour les variables qui ont des indices.

Ici, l' [sub](#)élément est utilisé pour représenter l'indice qui identifie la variable dans une famille de variables :

```
<p>The coordinate of the <var>i</var>th point is
```

```
(xi,
yi).
For example, the 10th point has coordinate
(x10, y10).
```

Les expressions mathématiques utilisent souvent des indices et des exposants. Les auteurs sont encouragés à utiliser MathML pour le balisage des mathématiques, mais les auteurs peuvent choisir d'utiliser [sub](#) et [sup](#) si un balisage mathématique détaillé n'est pas souhaité. [\[MATHML\]](#)

```
E=mc2
f(x, n) =
log4xn
```

#### 4.5.20 L' [i](#)élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML<sup>E</sup>lement](#).

L' [i](#)élément [représente](#) une étendue de texte dans une voix ou une humeur alternative, ou autrement décalée par rapport à la prose normale d'une manière indiquant une qualité de texte différente, telle qu'une désignation taxonomique, un terme technique, une phrase idiomatique d'une autre langue, une translittération, une pensée, ou un nom de navire dans les textes occidentaux.

Les termes dans des langues différentes du texte principal doivent être annotés avec [lang](#) des attributs (ou, en XML, [lang](#) [des attributs dans l' espace de noms XML](#) ).

Les exemples ci-dessous montrent les utilisations de l' [i](#) élément :

```
<p>The <i class="taxonomy">Felis silvestris catus</i> is cute.</p>
<p>The term <i>prose content</i> is defined above.</p>
<p>There is a certain <i lang="fr">je ne sais quoi</i> in the
air.</p>
```

Dans l'exemple suivant, une séquence de rêve est balisée à l'aide [i](#) d'éléments.

```
<p>Raymond tried to sleep.</p>
<p><i>The ship sailed away on Thursday</i>, he
dreamt. <i>The ship had many people aboard, including a beautiful
princess called Carey. He watched her, day-in, day-out, hoping she
would notice him, but she never did.</i></p>
<p><i>Finally one night he picked up the courage to speak with
her-</i></p>
<p>Raymond woke with a start as the fire alarm rang out.</p>
```

Les auteurs peuvent utiliser l' [class](#) attribut sur l' [i](#) élément pour identifier pourquoi l'élément est utilisé, de sorte que si le style d'une utilisation particulière (par exemple, les séquences de rêve par opposition aux termes taxonomiques) doit être modifié ultérieurement, l'auteur ne le fasse pas. doivent parcourir l'intégralité du document (ou une série de documents connexes) en annotant chaque utilisation.

Les auteurs sont encouragés à considérer si d'autres éléments pourraient être plus applicables que l' [i](#) élément, par exemple l' [em](#) élément pour marquer l'emphase de l'accent, ou l' [dfn](#) élément pour marquer l'instance de définition d'un terme.

*Les feuilles de style peuvent être utilisées pour formater [i](#) les éléments, tout comme n'importe quel autre élément peut être restylé. Ainsi, il n'est pas vrai que le contenu des [i](#) éléments sera nécessairement en italique.*

#### 4.5.21 L' [b](#) élément



##### [Catégories](#) :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu du phrasé est attendu.

### Modèle de contenu :

Contenu de la phrase .

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

Attributs globaux

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

Utilisations HTML`Element`.

L'élément b représente une étendue de texte sur laquelle l'attention est attirée à des fins utilitaires sans transmettre d'importance supplémentaire et sans implication d'une voix ou d'une humeur alternative, comme des mots clés dans un résumé de document, des noms de produits dans une revue, des mots exploitables dans logiciel interactif piloté par texte, ou un article dirigé.

L'exemple suivant montre une utilisation de l'élément b pour mettre en évidence des mots clés sans les marquer comme importants :

```
<p>The <b>frobonitor</b> and <b>barbinator</b> components are  
fried.</p>
```

In the following example, objects in a text adventure are highlighted as being special by use of the b element.

```
<p>You enter a small room. Your <b>sword</b> glows  
brighter. A <b>rat</b> scurries past the corner wall.</p>
```

Another case where the b element is appropriate is in marking up the lede (or lead) sentence or paragraph. The following example shows how a BBC article about kittens adopting a rabbit as their own could be marked up:

```
<article>  
  <h2>Kittens 'adopted' by pet rabbit</h2>  
  <p><b class="lede">Six abandoned kittens have found an  
  unexpected new mother figure – a pet rabbit.</b></p>  
  <p>Veterinary nurse Melanie Humble took the three-week-old  
  kittens to her Aberdeen home.</p>  
  [...]
```

As with the [i](#) element, authors can use the [class](#) attribute on the [b](#) element to identify why the element is being used, so that if the style of a particular use is to be changed at a later date, the author doesn't have to go through annotating each use.

L' [b](#)élément doit être utilisé en dernier recours lorsqu'aucun autre élément n'est plus approprié. En particulier, les titres doivent utiliser les éléments [h1](#) to [h6](#), l'emphase doit utiliser l' [em](#)élément, l'importance doit être indiquée avec l' [strong](#)élément et le texte marqué ou mis en évidence doit utiliser l' [mark](#) élément.

Ce qui suit serait une utilisation *incorrecte* :

```
<p><b>WARNING!</b> Do not frob the barbinator!</p>
```

Dans l'exemple précédent, l'élément correct à utiliser aurait été [strong](#), et non [b](#).

*Les feuilles de style peuvent être utilisées pour formater [b](#)les éléments, tout comme n'importe quel autre élément peut être restylé. Ainsi, il n'est pas vrai que le contenu des [b](#)éléments sera nécessairement mis en gras.*

#### 4.5.22 L' [u](#)élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTML](#)[Element](#).

L' uélément [représente](#) une étendue de texte avec une annotation non textuelle non articulée, bien que rendue explicitement, telle que l'étiquetage du texte comme étant un nom propre dans le texte chinois (une marque de nom propre chinois) ou l'étiquetage du texte comme étant mal orthographié.

Dans la plupart des cas, un autre élément est susceptible d'être plus approprié : pour marquer l'accent mis sur l'accentuation, l' emélément doit être utilisé ; pour marquer des mots ou des phrases clés, l' bélément ou l' markélément doit être utilisé, selon le contexte ; pour marquer les titres de livres, l' citeélément doit être utilisé ; pour étiqueter le texte avec des annotations textuelles explicites, l' rubyélément doit être utilisé ; pour les termes techniques, la désignation taxonomique, la translittération, une pensée ou pour l'étiquetage des noms de navires dans les textes occidentaux, l' i élément doit être utilisé.

*Le rendu par défaut de l' uélément dans les présentations visuelles se heurte au rendu conventionnel des hyperliens (souligné). Les auteurs sont encouragés à éviter d'utiliser l' uélément où il pourrait être confondu avec un hyperlien.*

Dans cet exemple, un uélément est utilisé pour marquer un mot comme mal orthographié :

```
<p>The <u>see</u> is full of fish.</p>
```

#### 4.5.23 L' markélément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

## Utilisations HTMLElement.

L' markélément représente une séquence de texte dans un document marqué ou mis en évidence à des fins de référence , en raison de sa pertinence dans un autre contexte. Lorsqu'il est utilisé dans une citation ou un autre bloc de texte référencé dans la prose, il indique un surlignage qui n'était pas présent à l'origine mais qui a été ajouté pour attirer l'attention du lecteur sur une partie du texte qui n'aurait peut-être pas été considérée comme importante par le auteur original lorsque le bloc a été écrit à l'origine, mais qui fait maintenant l'objet d'un examen minutieux auparavant inattendu. Lorsqu'il est utilisé dans la prose principale d'un document, il indique une partie du document qui a été mise en évidence en raison de sa pertinence probable pour l'activité actuelle de l'utilisateur.

Cet exemple montre comment l' markélément peut être utilisé pour attirer l'attention sur une partie particulière d'une citation :

```
<p lang="en-US">Consider the following quote:</p>
<blockquote lang="en-GB">
  <p>Look around and you will find, no-one's really
  <mark>colour</mark> blind.</p>
</blockquote>
<p lang="en-US">As we can tell from the <em>spelling</em> of the
word,
the person writing this quote is clearly not American.</p>
```

(Si le but était de marquer l'élément comme mal orthographié, cependant, l' uélément, éventuellement avec une classe, serait plus approprié.)

Un autre exemple de l' markélément met en surbrillance les parties d'un document qui correspondent à une chaîne de recherche. Si quelqu'un regardait un document et que le serveur savait que l'utilisateur cherchait le mot "chaton", alors le serveur pourrait renvoyer le document avec un paragraphe modifié comme suit :

```
<p>I also have some <mark>kitten</mark>s who are visiting me
these days. They're really cute. I think they like my garden! Maybe
I
should adopt a <mark>kitten</mark>.</p>
```

Dans l'extrait de code suivant, un paragraphe de texte fait référence à une partie spécifique d'un fragment de code.

```
<p>The highlighted part below is where the error lies:</p>
<pre><code>var i: Integer;
begin
  i := <mark>1.1</mark>;
end.</code></pre>
```



Ceci est distinct de *la coloration syntaxique*, qui span est plus appropriée. En combinant les deux, on obtiendrait :

```
<p>The highlighted part below is where the error lies:</p>
<pre><code><span class=keyword>var</span> <span
class=ident>i</span>: <span class=type>Integer</span>;
<span class=keyword>begin</span>
  <span class=ident>i</span> := <span
class=literal><mark>1.1</mark></span>;
<span class=keyword>end</span>.</code></pre>
```

Ceci est un autre exemple montrant l'utilisation de mark pour mettre en surbrillance une partie du texte cité qui n'était pas mise en évidence à l'origine. Dans cet exemple, les conventions typographiques courantes ont conduit l'auteur à styliser explicitement mark les éléments entre guillemets pour les afficher en italique.

```
<style>
  blockquote mark, q mark {
    font: inherit; font-style: italic;
    text-decoration: none;
    background: transparent; color: inherit;
  }
  .bubble em {
    font: inherit; font-size: larger;
    text-decoration: underline;
  }
</style>
<article>
  <h1>She knew</h1>
  <p>Did you notice the subtle joke in the joke on panel 4?</p>
  <blockquote>
    <p class="bubble">I didn't <em>want</em> to believe. <mark>Of
course
    on some level I realized it was a known-plaintext attack.</mark>
But I
    couldn't admit it until I saw for myself.</p>
  </blockquote>
  <p>(Emphasis mine.) I thought that was great. It's so pedantic,
yet it
  explains everything neatly.</p>
</article>
```

Notez, incidemment, la distinction entre l' em élément dans cet exemple, qui fait partie du texte original cité, et l' mark élément, qui met en évidence une partie pour un commentaire.

L'exemple suivant montre la différence entre indiquer l' *importance* d'une étendue de texte ( strong) et indiquer la *pertinence* d'une étendue de texte ( mark). Il s'agit d'un extrait d'un manuel, où l'extrait a mis en évidence les parties pertinentes pour l'examen. Les avertissements de sécurité, aussi importants soient-ils, ne sont apparemment pas pertinents pour l'examen.

```
<h3>Wormhole Physics Introduction</h3>

<p><mark>A wormhole in normal conditions can be held open for a
maximum of just under 39 minutes.</mark> Conditions that can
increase
the time include a powerful energy source coupled to one or both of
the gates connecting the wormhole, and a large gravity well (such
as a
black hole).</p>

<p><mark>Momentum is preserved across the wormhole. Electromagnetic
radiation can travel in both directions through a wormhole,
but matter cannot.</mark></p>

<p>When a wormhole is created, a vortex normally forms.
<strong>Warning: The vortex caused by the wormhole opening will
annihilate anything in its path.</strong> Vortexes can be avoided
when
using sufficiently advanced dialing technology.</p>

<p><mark>An obstruction in a gate will prevent it from accepting a
wormhole connection.</mark></p>
```

#### 4.5.24 L' **bd**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

Attributs globaux

De plus, l' dir attribut global a une sémantique spéciale sur cet élément.

## Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

## Interface DOM :

Utilisations HTMLElement.

L' bdi élément représente une étendue de texte qui doit être isolée de son environnement aux fins de formatage de texte bidirectionnel. [BIDI]

*L' dir attribut global est par défaut auto sur cet élément (il n'hérite jamais de l'élément parent comme avec les autres éléments).*

*Cet élément a des exigences de rendu impliquant l'algorithme bidirectionnel .*

Cet élément est particulièrement utile lors de l'intégration de contenu généré par l'utilisateur avec une directionnalité inconnue.

Dans cet exemple, les noms d'utilisateur sont affichés avec le nombre de messages que l'utilisateur a soumis. Si l' bdi élément n'était pas utilisé, le nom d'utilisateur de l'utilisateur arabe finirait par confondre le texte (l'algorithme bidirectionnel mettrait les deux-points et le chiffre "3" à côté du mot "Utilisateur" plutôt qu'à côté du mot "messages") .

```
<ul>
  <li>User <bdi>jcranmer</bdi>: 12 posts.
  <li>User <bdi>hober</bdi>: 5 posts.
  <li>User <bdi>إيان</bdi>: 3 posts.
</ul>
```

- User jcranmer: 12 posts.
- User hober: 5 posts.
- User إيان: 3 posts.

Lors de l'utilisation de l' bdi élément, le nom

- User jcranmer: 12 posts.
- User hober: 5 posts.
- User 3 إيان posts.

d'utilisateur agit comme prévu.

Si l' bdi élément devait

être remplacé par un b élément, le nom d'utilisateur confondrait l'algorithme bidirectionnel et la troisième puce finirait par dire "Utilisateur 3 :", suivi du nom arabe (de droite à gauche), suivi de "messages" et une période.

#### 4.5.25 L' **bdo**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Contenu de la phrase](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
De plus, l' [dir](#)attribut global a une sémantique spéciale sur cet élément.

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTMLElement](#).

L' [bdo](#)élément [représente](#) le contrôle de formatage explicite de la directionnalité du texte pour ses enfants. Il permet aux auteurs de remplacer l'algorithme bidirectionnel Unicode en spécifiant explicitement un remplacement de direction. [\[BIDI\]](#)

Les auteurs doivent spécifier l' [dir](#)attribut sur cet élément, avec la valeur [ltr](#)pour spécifier un remplacement de gauche à droite et avec la valeur [rtl](#)pour spécifier un remplacement de droite à gauche. La [auto](#)valeur ne doit pas être spécifiée.

*Cet élément [a des exigences de rendu impliquant l'algorithme bidirectionnel](#) .*

#### 4.5.26 L' **span**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .

Contenu palpable .

**Contextes dans lesquels cet élément peut être utilisé :**

Où le contenu du phrasé est attendu.

**Modèle de contenu :**

Contenu de la phrase .

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

**Attributs de contenu :**

## Attributs globaux

**Considérations d'accessibilité :**

[Pour les auteurs](#) .

Pour les exécutants :

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLSpanElement : HTMLElement {
```

```
[HTMLConstructor] constructor();
```

} ;

L' span élément ne signifie rien en lui-même, mais peut être utile lorsqu'il est utilisé avec les attributs globaux , par exemple class, lang ou dir. Il représente ses enfants.

Dans cet exemple, un fragment de code est balisé à l'aide spand'éléments et classd'attributs afin que ses mots clés et identifiants puissent être codés par couleur à partir de CSS :

```
<pre><code class="lang-c"><span class="keyword">for</span> (<span class="ident">j</span> = 0; <span class="ident">j</span> < 256; <span class="ident">j</span>++) {  
    <span class="ident">i_t3</span> = (<span class="ident">i_t3</span> & 0x1ffff) | (<span class="ident">j</span> < & & 17);  
    <span class="ident">i_t6</span> = ((((((<span class="ident">i_t3</span> >> 3) ^ <span class="ident">i_t3</span>) >> 1) ^ <span class="ident">i_t3</span>) >> 8) ^ <span class="ident">i_t3</span>) >> 5) & 0xff;  
    <span class="keyword">if</span> (<span class="ident">i_t6</span> == <span class="ident">i_t1</span>)  
        <span class="keyword">break</span>;  
}</code></pre>
```

#### 4.5.27 L' **br**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Rien](#) .

##### Omission de balise dans text/html :

Pas [de balise de fin](#) .

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLBRElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    // also has obsolete members
```

```
};
```

L' **br**élément [représente](#) un saut de ligne.

*Alors que les sauts de ligne sont généralement représentés dans les médias visuels en déplaçant physiquement le texte suivant vers une nouvelle ligne, une feuille de style ou un agent utilisateur serait également justifié de faire en sorte que les sauts de ligne soient rendus d'une manière différente, par exemple sous forme de points verts, ou espacement.*

**br**Les éléments doivent être utilisés uniquement pour les sauts de ligne qui font réellement partie du contenu, comme dans les poèmes ou les adresses.

L'exemple suivant est l'utilisation correcte de l'brélément :

```
<p>P. Sherman<br>
42 Wallaby Way<br>
Sydney</p>
```

brles éléments ne doivent pas être utilisés pour séparer les groupes thématiques dans un paragraphe.

Les exemples suivants ne sont pas conformes, car ils abusent de l'brélément :

```
<p><a ...>34 comments.</a><br>
<a ...>Add a comment.</a></p>
<p><label>Name: <input name="name"></label><br>
<label>Address: <input name="address"></label></p>
```

Voici des alternatives à ce qui précède, qui sont correctes :

```
<p><a ...>34 comments.</a></p>
<p><a ...>Add a comment.</a></p>
<p><label>Name: <input name="name"></label></p>
<p><label>Address: <input name="address"></label></p>
```

Si un paragraphe n'est constitué que d'un seul brélément, il représente une ligne vierge de substitution (par exemple, comme dans un modèle). Ces lignes vides ne doivent pas être utilisées à des fins de présentation.

Tout contenu à l'intérieur brdes éléments ne doit pas être considéré comme faisant partie du texte environnant.

*Cet élément a des exigences de rendu impliquant l'algorithme bidirectionnel .*

#### 4.5.28 L' wbrélément



##### Catégories :

Contenu du flux .  
Contenu de la phrase .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu du phrasé est attendu.

##### Modèle de contenu :

Rien .

### Omission de balise dans text/html :

Pas de balise de fin .

### Attributs de contenu :

Attributs globaux

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

Utilisations HTMLElement.

L' wbrélément représente une opportunité de saut de ligne.

Dans l'exemple suivant, quelqu'un est cité comme disant quelque chose qui, pour effet, est écrit comme un long mot. Cependant, pour s'assurer que le texte peut être enveloppé de manière lisible, les mots individuels de la citation sont séparés à l'aide d'un wbrélément.

```
<p>So then she pointed at the tiger and screamed  
"there<wbr>is<wbr>no<wbr>way<wbr>you<wbr>are<wbr>ever<wbr>going<wbr>  
>to<wbr>catch<wbr>me"!</p>
```

Tout contenu à l'intérieur wbrdes éléments ne doit pas être considéré comme faisant partie du texte environnant.

```
var wbr = document.createElement("wbr");  
wbr.textContent = "This is wrong";  
document.body.appendChild(wbr);
```

Cet élément a des exigences de rendu impliquant l'algorithme bidirectionnel .

## 4.5.29 Résumé d'utilisation

*Cette section est non normative.*

Élément	But	Exemple
<u>a</u>	Hyperliens	Visit my <code>&lt;a href="drinks.html"&gt;drinks&lt;/a&gt;</code> page.
<u>em</u>	Accent mis sur le stress	I must say I <code>&lt;em&gt;adore&lt;/em&gt;</code> lemonade.
<u>strong</u>	Importance	This tea is <code>&lt;strong&gt;very hot&lt;/strong&gt;</code> .
<u>small</u>	Commentaires annexes	These grapes are made into wine. <code>&lt;small&gt;Alcohol is addictive.&lt;/small&gt;</code>
<u>s</u>	Texte inexact	Price: <code>&lt;s&gt;£4.50&lt;/s&gt;</code> £2.00!



Élément	But	Exemple
<u><a href="#">cite</a></u>	Titres d'oeuvres	The case <code>&lt;cite&gt;Hugo v. Danielle&lt;/cite&gt;</code> is relevant here.
<u><a href="#">q</a></u>	Citations	The judge said <code>&lt;q&gt;You can drink water from the fish tank&lt;/q&gt;</code> but advised against it.
<u><a href="#">dfn</a></u>	Définition de l'instance	The term <code>&lt;dfn&gt;organic food&lt;/dfn&gt;</code> refers to food produced without synthetic chemicals.
<u><a href="#">abbr</a></u>	Abréviations	Organic food in Ireland is certified by the <code>&lt;abbr title="Irish Organic Farmers and Growers Association"&gt;IOFGA&lt;/abbr&gt;</code> .
<u><a href="#">ruby</a></u> , <u><a href="#">rt</a></u> , <u><a href="#">rp</a></u>	Annotations rubis	<code>&lt;ruby&gt; OJ &lt;rp&gt;(&lt;rt&gt;Orange Juice&lt;rp&gt;&lt;/ruby&gt;</code>
<u><a href="#">data</a></u>	Équivalent lisible par machine	Available starting today! <code>&lt;data value="UPC:022014640201"&gt;North Coast Organic Apple Cider&lt;/data&gt;</code>
<u><a href="#">time</a></u>	Équivalent lisible par machine des données relatives à la date ou à l'heure	Available starting on <code>&lt;time datetime="2011-11-18"&gt;November 18th&lt;/time&gt;!</code>
<u><a href="#">code</a></u>	Code informatique	The <code>&lt;code&gt;fruitdb&lt;/code&gt;</code> program can be used for tracking fruit production.
<u><a href="#">var</a></u>	variables	If there are <code>&lt;var&gt;n&lt;/var&gt;</code> fruit in the bowl, at least <code>&lt;var&gt;n&lt;/var&gt;+2</code> will be ripe.
<u><a href="#">samp</a></u>	Sortie informatique	The computer said <code>&lt;samp&gt;Unknown error - 3&lt;/samp&gt;</code> .
<u><a href="#">kbd</a></u>	Entrée utilisateur	Hit <code>&lt;kbd&gt;F1&lt;/kbd&gt;</code> to continue.
<u><a href="#">sub</a></u>	Indices	Water is H <code>&lt;sub&gt;2&lt;/sub&gt;</code> O.
<u><a href="#">sup</a></u>	Exposants	The Hydrogen in heavy water is usually <code>&lt;sup&gt;2&lt;/sup&gt;</code> H.
<u><a href="#">i</a></u>	Voix alternative	Lemonade consists primarily of <code>&lt;i&gt;Citrus limon&lt;/i&gt;</code> .
<u><a href="#">b</a></u>	Mots clés	Take a <code>&lt;b&gt;lemon&lt;/b&gt;</code> and squeeze it with a <code>&lt;b&gt;juicer&lt;/b&gt;</code> .
<u><a href="#">u</a></u>	Remarques	The mixture of apple juice and <code>&lt;u class="spelling"&gt;elderflower&lt;/u&gt;</code> juice is very pleasant.
<u><a href="#">mark</a></u>	Souligner	Elderflower cordial, with one <code>&lt;mark&gt;part&lt;/mark&gt;</code> cordial to ten <code>&lt;mark&gt;part&lt;/mark&gt;</code> s water, stands a <code>&lt;mark&gt;part&lt;/mark&gt;</code> from the rest.
<u><a href="#">bdi</a></u>	Isolation de la directionnalité du texte	The recommended restaurant is <code>&lt;bdi lang=""&gt;My Juice Café (At The Beach)&lt;/bdi&gt;</code> .
<u><a href="#">bdo</a></u>	Formatage de la directionnalité du texte	The proposal is to write English, but in reverse order. "Juice" would become " <code>&lt;bdo dir=rtl&gt;Juice&lt;/bdo&gt;</code> "
<u><a href="#">span</a></u>	Autre	In French we call it <code>&lt;span lang="fr"&gt;sirop de sureau&lt;/span&gt;</code> .

Élément	But	Exemple
<a href="#"><u>br</u></a>	Saut de ligne	Simply Orange Juice Company Apopka, FL 32703 U.S.A.
<a href="#"><u>wbr</u></a>	Opportunité de rupture de ligne	www.simply<wbr>orange<wbr>juice.com

## 4.6 Liens

### 4.6.1 Présentation

Les liens sont une construction conceptuelle, créée par les éléments [a](#), [area](#), [form](#) et [link](#), qui [représentent](#) une connexion entre deux ressources, dont l'une est le courant [Document](#). Il existe deux types de liens en HTML :

#### **Liens vers des ressources externes**

Ce sont des liens vers des ressources qui doivent être utilisées pour enrichir le document courant, généralement traitées automatiquement par l'agent utilisateur. Tous [les liens de ressources externes](#) ont un [processus d'extraction et de traitement de l'](#) algorithme de ressource lié qui décrit comment la ressource est obtenue.

#### **Hyperliens**

Ce sont des liens vers d'autres ressources qui sont généralement exposées à l'utilisateur par l'agent utilisateur afin que l'utilisateur puisse amener l'agent utilisateur à [naviguer](#) vers ces ressources, par exemple pour les visiter dans un navigateur ou les télécharger.

Pour [link](#) les éléments avec un [href](#) attribut et un [rel](#) attribut, des liens doivent être créés pour les mots-clés de l' [rel](#) attribut, comme défini pour ces mots-clés dans la section [des types de liens](#) .

De même, pour [a](#) et [area](#) les éléments avec un [href](#) attribut et un [rel](#) attribut, des liens doivent être créés pour les mots-clés de l' [rel](#) attribut tel que défini pour ces mots-clés dans la section [des types de liens](#) . [link](#) Cependant, contrairement aux éléments [a](#) et [area](#) aux éléments avec un [href](#) attribut qui soit n'ont pas d' [rel](#) attribut, soit dont [rel](#) l'attribut n'a pas de mots-clés définis comme spécifiant [des hyperliens](#) , doivent également créer un [hyperlien](#) . Ce lien hypertexte implicite n'a pas de signification particulière (il n'a pas [de type de lien](#) ) au-delà de la liaison du [document de nœud](#) de l'élément à la ressource donnée par le document de l'élément. [href](#) attribut.

De même, pour [form](#) les éléments avec un [rel](#) attribut, des liens doivent être créés pour les mots-clés de l' [rel](#) attribut tels que définis pour ces mots-clés dans la

section [des types de liens](#) . [form](#)les éléments qui n'ont pas d' [rel](#)attribut, ou dont [rel](#)l'attribut n'a pas de mots-clés définis comme spécifiant [des hyperliens](#) , doivent également créer un [hyperlien](#) .

Un [lien hypertexte](#) peut avoir une ou plusieurs **annotations de lien hypertexte** qui modifient la sémantique de traitement de ce lien hypertexte.

#### 4.6.2 Liens créés par [a](#)et [area](#)éléments

L' [href](#) attribut sur les éléments [a](#)et [area](#)doit avoir une valeur qui est une [URL valide potentiellement entourée d'espaces](#) .

*L' [href](#)attribut on [a](#)and [area](#)elements n'est pas obligatoire ; lorsque ces éléments n'ont pas [href](#)d'attributs, ils ne créent pas d'hyperliens.*

L' [target](#) attribut, s'il est présent, doit être un [nom de cible navigable valide ou un mot-clé](#) . Il donne le nom du [navigable](#) qui sera utilisé. Les agents utilisateurs utilisent ce nom lorsqu'ils [suivent des hyperliens](#) .

Lorsqu'un [comportement d'activation](#) d'un élément [a](#)ou est invoqué, l'agent utilisateur peut permettre à l'utilisateur d'indiquer une préférence concernant si l'hyperlien doit être utilisé pour [la navigation](#) ou si la ressource qu'il spécifie doit être téléchargée.[area](#)

En l'absence de préférence de l'utilisateur, la valeur par défaut doit être la navigation si l'élément n'a pas [download](#)d'attribut, et doit être de télécharger la ressource spécifiée si c'est le cas.

Que ce soit déterminé par les préférences de l'utilisateur ou via la présence ou l'absence de l'attribut, si la décision est d'utiliser l'hyperlien pour la [navigation](#) , alors l'agent utilisateur doit [suivre l'hyperlien](#) , et si la décision est d'utiliser l'hyperlien pour télécharger une ressource, le l'agent utilisateur doit [télécharger le lien hypertexte](#) . Ces termes sont définis dans les sections suivantes ci-dessous.

L' [download](#) attribut, s'il est présent, indique que l'auteur souhaite que le lien hypertexte soit utilisé pour [télécharger une ressource](#) . L'attribut peut avoir une valeur ; la valeur, le cas échéant, spécifie le nom de fichier par défaut que l'auteur recommande d'utiliser pour étiqueter la ressource dans un système de fichiers local. Il n'y a pas de restrictions sur les valeurs autorisées, mais les auteurs sont avertis que la plupart des systèmes de fichiers ont des limitations en ce qui concerne la ponctuation prise en charge dans les noms de fichiers, et les agents utilisateurs sont susceptibles d'ajuster les noms de fichiers en conséquence.

L' **ping**attribut, s'il est présent, donne les URL des ressources qui souhaitent être notifiées si l'utilisateur suit l'hyperlien. La valeur doit être un [ensemble de jetons séparés par des espaces](#) , chacun d'entre eux devant être une [URL non vide valide](#) dont [le schéma](#) est un [schéma HTTP\(S\)](#) . La valeur est utilisée par l'agent utilisateur pour [l'audit des liens hypertexte](#) .

L' **rel**attribut on [a](#)et [area](#) elements contrôle les types de liens créés par les éléments. La valeur de l'attribut doit être un [ensemble non ordonné de jetons uniques séparés par des espaces](#) . Les [mots-clés autorisés et leurs significations](#) sont définis ci-dessous.

**rel**Les [jetons pris en charge par](#) sont les mots-clés définis dans [les types de liens HTML](#) qui sont autorisés sur les éléments [a](#)et [area](#) , ont un impact sur le modèle de traitement et sont pris en charge par l'agent utilisateur. [Les jetons pris en charge](#) possibles sont [noreferrer](#), [noopener](#)et [opener](#). **rel**Les [jetons pris en charge par](#) doivent uniquement inclure les jetons de cette liste pour lesquels l'agent utilisateur implémente le modèle de traitement.

L' **rel**attribut n'a pas de valeur par défaut. Si l'attribut est omis ou si aucune des valeurs de l'attribut n'est reconnue par l'agent utilisateur, alors le document n'a pas de relation particulière avec la ressource de destination autre qu'il y a un lien hypertexte entre les deux.

L' **hreflang** attribut sur [a](#)les éléments qui créent [des hyperliens](#) , s'il est présent, donne la langue de la ressource liée. C'est purement consultatif. La valeur doit être une balise de langue BCP 47 valide. [\[BCP47\]](#) Les agents utilisateurs ne doivent pas considérer cet attribut comme faisant autorité — lors de la récupération de la ressource, les agents utilisateurs doivent utiliser uniquement les informations de langue associées à la ressource pour déterminer sa langue, et non les métadonnées incluses dans le lien vers la ressource.

L' **type** attribut, s'il est présent, donne le [type MIME](#) de la ressource liée. C'est purement consultatif. La valeur doit être une [chaîne de type MIME valide](#) . Les agents utilisateurs ne doivent pas considérer l' **type**attribut comme faisant autorité - lors de la récupération de la ressource, les agents utilisateurs ne doivent pas utiliser les métadonnées incluses dans le lien vers la ressource pour déterminer son type.

L' **referrerpolicy**attribut est un [attribut de stratégie de référence](#) . Son but est de définir la [politique de référence](#) utilisée lors [du suivi des hyperliens](#) . [\[POLITIQUE DE RÉFÉRENCE\]](#)

#### 4.6.3 API pour [a](#)et [area](#)éléments

```
interface mixin HTMLHyperlinkElementUtils {
```

```
[CEReactions] stringifier attribute USVString href;
```

```
readonly attribute USVString origin;
```

```
[CEReactions] attribute USVString protocol;
```

```
[CEReactions] attribute USVString username;
```

```
[CEReactions] attribute USVString password;
```

```
[CEReactions] attribute USVString host;
```

```
[CEReactions] attribute USVString hostname;
```

```
[CEReactions] attribute USVString port;
```

```
[CEReactions] attribute USVString pathname;
```

```
[CEReactions] attribute USVString search;
```

```
[CEReactions] attribute USVString hash;
```

```
};
```

**`hyperlink.toString()`**

**`hyperlink.href`**

✓

Renvoie l'URL du lien hypertexte.

Peut être défini, pour changer l'URL.

**`hyperlink.origin`**

✓

Renvoie l'origine de l'URL du lien hypertexte.

**`hyperlink.protocol`**

✓

Renvoie le schéma de l'URL du lien hypertexte.

Peut être défini pour modifier le schéma de l'URL.

**`hyperlink.username`**

✓

Renvoie le nom d'utilisateur de l'URL du lien hypertexte.

Peut être défini pour modifier le nom d'utilisateur de l'URL.

`hyperlink.password`

✓ 

Renvoie le mot de passe de l'URL du lien hypertexte.

Peut être défini pour changer le mot de passe de l'URL.

`hyperlink.host`

✓ 

Renvoie l'hôte et le port de l'URL du lien hypertexte (s'il est différent du port par défaut du schéma).

Peut être défini pour modifier l'hôte et le port de l'URL.

`hyperlink.hostname`

✓ 

Renvoie l'hôte de l'URL du lien hypertexte.

Peut être défini pour changer l'hôte de l'URL.

`hyperlink.port`

✓ 

Renvoie le port de l'URL du lien hypertexte.

Peut être défini pour changer le port de l'URL.

`hyperlink.pathname`

✓ 

Renvoie le chemin de l'URL du lien hypertexte.

Peut être défini pour modifier le chemin de l'URL.

`hyperlink.search`

✓ 

Renvoie la requête de l'URL du lien hypertexte (inclut le "?" si non vide).

Peut être défini pour modifier la requête de l'URL (ignore le "?").

`hyperlink.hash`

✓ 

Renvoie le fragment d'URL du lien hypertexte (inclut le "#" si non vide).

Peut être défini, pour changer le fragment de l'URL (ignore le "#" de début).

Un élément implémentant le mixin a  
une `url` [HTMLHyperlinkElementUtils](#) associée (null ou une [URL](#)). Il est  
initialement nul.

Un élément implémentant le [HTMLHyperlinkElementUtils](#) mixin est associé à un **ensemble d'algorithmes d'url** , qui exécute ces étapes :

1. Si [href](#) l'attribut content de cet élément est absent, définissez [l'url](#) de cet élément sur null.
2. Sinon, analysez [href](#) la valeur de l'attribut content de cet élément par rapport au [nœud document](#) de cet élément . Si [l'analyse](#) réussit, définissez [l'URL](#) de cet élément sur le résultat ; sinon, définissez [l'url](#) de cet élément sur null.

Lorsque des éléments implémentant le [HTMLHyperlinkElementUtils](#) mixin sont créés, et chaque fois que ces éléments ont leur [href](#) attribut content défini, modifié ou supprimé, l'agent utilisateur doit [définir l'url](#) .

*Ceci n'est observable que pour [blob:](#) les URL, car leur [analyse](#) implique une recherche [dans le magasin d'URL Blob](#) .*

Un élément implémentant le [HTMLHyperlinkElementUtils](#) mixin est associé à un algorithme **de réinitialisation d'URL** , qui exécute ces étapes :

1. Si [l'URL](#) de l'élément n'est pas nulle, que son [schéma](#) est " `blob`" et qu'il a un [chemin opaque](#) , terminez ces étapes.
2. [Définissez l'url](#) .

Pour **mettre à jour**[href](#) , définissez la [href](#) valeur de l'attribut content de l'élément sur l' [url](#) de l'élément , [serialized](#) .

---

Les [href](#) étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Soit `url` l' [url](#) de [celle-ci](#) .
3. Si `url` est null et [n'a](#) pas [href](#) d'attribut de contenu, renvoie la chaîne vide.
4. Sinon, si `url` est null, renvoie la valeur de [cet href](#) attribut de contenu.
5. `URL` de retour , [sérialisée](#) .

Les [href](#) étapes du setter consistent à définir la valeur de [cet href](#) attribut de contenu sur la valeur donnée.

Les [origin](#) étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Si l'[URL](#) de [cet](#) est nulle, renvoie la chaîne vide.
3. Renvoie la [sérialisation](#) de l' [origine](#) de [cette URL](#) .

Les [protocol](#)étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Si [cette](#) URL [est](#) nulle, retournez " :".
3. Renvoie le [schéma de cette](#) URL , suivi de " " .:

Les [protocol](#)étapes du setter sont :

1. [Réinitialiser l'url](#) .
2. Si [cette](#) URL [est](#) nulle, alors retournez.
3. [L'URL de base analyse](#) la valeur donnée, suivie de " :", avec [cette](#) URL comme [URL](#) et [l'état de début du schéma](#) comme [remplacement d' état](#) .

*Étant donné que l'analyseur d'URL ignore plusieurs deux-points consécutifs, fournir une valeur de "https:" (ou même "https:::") équivaut à fournir une valeur de "https".*

4. [Mettre à jour](#)[href](#) .

Les [username](#)étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Si l'[URL](#) de [cet](#) est nulle, renvoie la chaîne vide.
3. Renvoie le nom d' [utilisateur](#) de [cette URL](#) .

Les [username](#)étapes du setter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* est null ou *url* [ne peut pas avoir de nom d'utilisateur/mot de passe/port](#) , alors retournez.
4. [Définissez le nom d'utilisateur](#) , l'[URL](#) donnée et la valeur donnée.
5. [Mettre à jour](#)[href](#) .



Les **password**étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* est null, renvoie la chaîne vide.
4. Renvoie [le mot de passe](#) de l'*url* .

Les **password**étapes du setter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* est null ou *url* [ne peut pas avoir de nom d'utilisateur/mot de passe/port](#) , alors retournez.
4. [Définissez le mot de passe](#) , l'*URL* donnée et la valeur donnée.
5. [Mettre à jour](#)[href](#) .

Les **host**étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* ou l'[hôte](#) de l' *url* est nul, renvoie la chaîne vide.
4. Si [le port](#) de l' *url* est nul, renvoie l'[hôte](#) de l' *url* , [sérialisé](#) .
5. Renvoie l'[hôte](#) de l' *url* , [sérialisé](#) , suivi de " " et [le port](#) de l' *url* , [sérialisé](#) . :

Les **host**étapes du setter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* est null ou *url* a un [chemin opaque](#) , alors retournez.
4. [L'URL de base analyse](#) la valeur donnée, avec *url* comme [url](#) et [host state](#) comme [state override](#) .
5. [Mettre à jour](#)[href](#) .

Les **hostname**étapes du getter sont :

1. [Réinitialiser l'url](#) .

2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* ou l'[hôte](#) de l' *url* est nul, renvoie la chaîne vide.
4. [Hôte](#) de l'*url* de retour , [sérialisé](#) .

Les [hostname](#) étapes du setter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* est null ou *url* a un [chemin opaque](#) , alors retournez.
4. [L'URL de base analyse](#) la valeur donnée, avec *url* comme [url](#) et [hostname](#) [state](#) comme [state override](#) .
5. [Mettre à jour](#)[href](#) .

Les [port](#) étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* ou [le port](#) de *url* est nul, renvoie la chaîne vide.
4. [Port](#) de l'*url* de retour , [sérialisé](#) .

Les [port](#) étapes du setter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* est null ou *url* [ne peut pas avoir de nom d'utilisateur/mot de passe/port](#) , alors retournez.
4. Si la valeur donnée est la chaîne vide, définissez [le port](#) de l' *url* sur null.
5. Sinon, [L'URL de base analyse](#) la valeur donnée, avec *url* comme [url](#) et [port](#) [state](#) comme [state override](#) .
6. [Mettre à jour](#)[href](#) .

Les [pathname](#) étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .

3. Si *url* est null, renvoie la chaîne vide.
4. Renvoie le résultat du [chemin d'URL sérialisant \*url\*](#).

Les [pathname](#)étapes du setter sont :

1. [Réinitialiser l'url](#).
2. Soit *url* l' [url](#) de [celle-ci](#).
3. Si *url* est null ou *url* a un [chemin opaque](#), alors retournez.
4. Définissez [le chemin](#) de l' *url* sur la liste vide.
5. [L'URL de base analyse](#) la valeur donnée, avec *url* comme [url](#) et [path start state](#) comme [state override](#).
6. [Mettre à jour](#)[href](#).

Les [search](#)étapes du getter sont :

1. [Réinitialiser l'url](#).
2. Soit *url* l' [url](#) de [celle-ci](#).
3. Si *url* est null, ou si [la requête](#) de *url* est null ou la chaîne vide, renvoie la chaîne vide.
4. Renvoie " ? " , suivi de [la requête](#) de l' *URL*.

Les [search](#)étapes du setter sont :

1. [Réinitialiser l'url](#).
2. Soit *url* l' [url](#) de [celle-ci](#).
3. Si *url* est null, terminez ces étapes.
4. Si la valeur donnée est la chaîne vide, définissez [la requête](#) de l' *url* sur null.
5. Sinon:
  1. Soit *l'entrée* la valeur donnée avec un seul " ? " supprimé, le cas échéant.
  2. Définissez [la requête](#) de l' *url* sur la chaîne vide.
  3. *Entrée d'analyse d'URL de base*, avec *url* comme [url](#) et [état de la requête](#) comme [remplacement d'état](#).
6. [Mettre à jour](#)[href](#).

Les **hash**étapes du getter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si *url* est null, ou si [le fragment](#) de *url* est null ou la chaîne vide, renvoie la chaîne vide.
4. Renvoie " # " , suivi du [fragment](#) d' *URL* .

Les **hash**étapes du setter sont :

1. [Réinitialiser l'url](#) .
2. Soit *url* l' [url](#) de [celle-ci](#) .
3. Si l'*URL* est nulle, alors retournez.
4. Si la valeur donnée est la chaîne vide, définissez [le fragment](#) de *url* sur null.
5. Sinon:
  1. Soit l'*entrée* la valeur donnée avec un seul " #" supprimé, le cas échéant.
  2. Définissez [le fragment](#) de l' *url* sur la chaîne vide.
  3. *Entrée d' [analyse d'URL de base](#)* , avec *url* comme [url](#) et [état du fragment](#) comme [remplacement d'état](#) .
6. [Mettre à jour](#)**href** .

#### 4.6.4 Suivre des hyperliens

Un *élément element* **ne peut pas naviguer** si l'une des conditions suivantes est vraie :

- [le document de nœud](#) de l'*élément* n'est pas [entièrement actif](#)
- *element* n'est pas un [a](#)élément et n'est pas [connecté](#) .

*Ceci est également utilisé par [la soumission de formulaire](#) pour l' [form](#)élément. L'exception pour [a](#)les éléments concerne la compatibilité avec le contenu Web.*

Pour **obtenir le noopener d'un élément** , étant donné un élément *élément*[a](#) , [area](#)ou [form](#)élément et une chaîne *cible* :

1. Si [les types de liens](#) de l' *élément* incluent le mot-clé ou , alors renvoie `true.noopenernoreferrer`
2. Si [les types de liens](#) de l' *élément* n'incluent pas le mot-clé et que la cible est une correspondance [ASCII non sensible à la casse](#) pour " ", alors renvoie `true.opener_blank`
3. Renvoie faux.

Pour **suivre le lien hypertexte** créé par un élément *subject* , étant donné un *hyperlinkSuffix* facultatif (null par défaut) :

1. Si le sujet [ne peut pas naviguer](#) , alors revenez.
2. Soit *remplacer* faux.
3. Laissez *targetAttributeValue* être la chaîne vide.
4. Si le sujet est un élément [a](#) ou [area](#), définissez *targetAttributeValue* sur le résultat de l'obtention de la cible donnée [d'un élément subject](#) .
5. Soit *noopener* le résultat de [l'obtention du noopener d'un élément](#) avec *subject* et *targetAttributeValue* .
6. Soit *targetNavigable* la première valeur de retour de l'application [des règles de choix d'un navigable](#) donné *targetAttributeValue* , le nœud du sujet [navigable](#) et *noopener* .
7. Si *targetNavigable* est null, alors retournez.
8. [Analyser une URL](#) en fonction de l'attribut du sujet `href` , par rapport au [nœud document](#) du sujet .
9. Si cela réussit, laissez *url* être la [chaîne d'URL résultante](#) .  
Sinon, si [l'analyse](#) de l' [URL](#) a échoué, revenez.
10. Si *hyperlinkSuffix* n'est pas nul, ajoutez-le à *url* .
11. Soit *referrerPolicy* l'état actuel de l'attribut content de *subject*`referrerpolicy` .
12. Si [les types de liens](#) du sujet incluent le mot-clé, définissez *referrerPolicy* sur " ".`noreferrerno-referrer`
13. [Naviguez entre](#) *targetNavigable* et *url* en utilisant [le document de nœud](#) du sujet , avec [referrerPolicy](#) défini sur *referrerPolicy* .

Contrairement à de nombreux autres types de navigation, suivre des hyperliens n'a pas de " [replace](#) " comportement spécial lorsque les documents ne sont pas [complètement chargés](#) . Cela est vrai pour les instances initiées par l'utilisateur des hyperliens suivants, ainsi que pour celles déclenchées par un script via, par exemple, `aElement.click()` .

## 4.6.5 Téléchargement des ressources



Dans certains cas, les ressources sont destinées à une utilisation ultérieure plutôt qu'à une consultation immédiate. Pour indiquer qu'une ressource est destinée à être téléchargée pour une utilisation ultérieure, plutôt qu'immédiatement utilisée, l'attribut `download` peut être spécifié sur l'élément `a` ou `area` qui crée le [lien hypertexte](#) vers cette ressource.

L'attribut peut en outre recevoir une valeur, pour spécifier le nom de fichier que les agents utilisateurs doivent utiliser lors du stockage de la ressource dans un système de fichiers. Cette valeur peut être remplacée par les paramètres `Content-Disposition` de nom de fichier de l'en-tête HTTP. [RFC6266]

Dans les situations d'origine croisée, l'attribut `download` doit être combiné avec l'en-tête `Content-Disposition`, en particulier avec le type `attachment` de disposition, pour éviter que l'utilisateur ne soit averti d'une activité potentiellement néfaste. (Il s'agit de protéger les utilisateurs contre le téléchargement d'informations personnelles ou confidentielles sensibles à leur insu.)

---

L'algorithme **de téléchargement autorisé** suivant prend deux booléens `sourceAllowsDownloading` et `targetAllowsDownloading`, et renvoie un booléen indiquant si le téléchargement est autorisé ou non :

1. Si `sourceAllowsDownloading` ou `targetAllowsDownloading` sont faux, alors renvoie faux.
2. Facultativement, l'agent utilisateur peut renvoyer false s'il pense que cela protégerait l'utilisateur d'un téléchargement potentiellement hostile.
3. Renvoie vrai.

Pour **télécharger le lien hypertexte** créé par un élément `subject`, étant donné un `hyperlinkSuffix` facultatif (null par défaut) :

1. Si le sujet [ne peut pas naviguer](#), alors revenez.
2. Soit `sourceAllowsDownloading` la valeur false si [l'indicateur de sandboxing actif](#) du [document de nœud](#) du sujet a l'[indicateur de contexte de navigation des téléchargements en sandbox](#) défini ; autrement vrai.
3. Si le résultat de l'algorithme [de téléchargement autorisé](#) avec `sourceAllowsDownloading` et vrai est faux, alors retournez.

4. [Analyser une URL](#) en fonction de l'attribut du *sujet* [href](#) , par rapport au [noeud document](#) du *sujet* .
5. Si [l'analyse de l'URL](#) échoue, revenez.
6. Sinon, laissez *URL* être la [chaîne d'URL résultante](#) .
7. Si *hyperlinkSuffix* n'est pas nul, ajoutez-le à *URL* .
8. Exécutez ces étapes [en parallèle](#) :
  1. Soit *request* une nouvelle [requête](#) dont [l'URL](#) est *URL* , [le client](#) est [l'objet de paramètres d'entrée](#) , [l'initiateur](#) est " download", [la destination](#) est la chaîne vide et dont [l'indicateur synchrone](#) et [l'indicateur use-URL-credentials](#) sont définis.
  2. Gérez le résultat de *la requête* [de récupération comme un téléchargement](#) .

Lorsqu'un agent utilisateur doit gérer une ressource obtenue à partir d'une récupération **en tant que téléchargement** , il doit fournir à l'utilisateur un moyen de sauvegarder la ressource pour une utilisation ultérieure, si une ressource est obtenue avec succès. Sinon, il doit signaler tout problème de téléchargement du fichier à l'utilisateur.

Si l'agent utilisateur a besoin d'un nom de fichier pour une ressource gérée [comme un téléchargement](#) , il doit en sélectionner un en utilisant l'algorithme suivant.

**Cet algorithme est destiné à atténuer les risques de sécurité liés au téléchargement de fichiers à partir de sites non fiables, et les agents utilisateurs sont fortement invités à le suivre.**

1. Soit *filename* la valeur indéfinie.
2. Si la ressource a un en- [Content-Disposition](#) tête `` , cet en-tête spécifie le `attachment` type de disposition et l'en-tête inclut des informations sur le nom de fichier, alors laissez *filename* avoir la valeur spécifiée par l'en-tête et passez à l'étape étiquetée *assainir* ci-dessous. [\[RFC6266\]](#)
3. Soit *l'origine de l'interface* l' [origine](#) de [Document](#) dans laquelle l' action [de téléchargement](#) ou [de navigation](#) résultant du téléchargement a été lancée, le cas échéant.
4. Soit *l'origine de la ressource* être l' [origine](#) de l'URL de la ressource en cours de téléchargement, sauf si le composant de [schéma](#)<sub>data</sub> de cette URL est , auquel cas l' *origine de la ressource* doit être la même que l' *origine de l'interface* , le cas échéant.
5. S'il n'y a pas d'*origine d'interface* , laissez *l'opération de confiance* être vraie. Sinon, laissez *l'opération de confiance* être vraie si *l'origine de la ressource* est la [même origine](#) que *l'origine de l'interface* , et fausse sinon.

6. Si l'opération de confiance est vraie et que la ressource a un Content-Disposition en-tête `` et que cet en-tête inclut des informations sur le nom de fichier, laissez *le nom de fichier* avoir la valeur spécifiée par l'en-tête et passez à l'étape intitulée *assainir* ci-dessous. [\[RFC6266\]](#)
7. Si le téléchargement n'a pas été initié à partir d'un lien hypertexte créé par un élément area, ou si l'élément de l' hyperlien à partir duquel il a été initié n'avait pas d' download attribut lorsque le téléchargement a été initié, ou s'il existait un tel attribut mais sa valeur lorsque le téléchargement a été lancé était la chaîne vide, puis passez à l'étape étiquetée *aucun nom de fichier proposé*.
8. Laissez *le nom de fichier proposé* avoir la valeur de l' download attribut de l'élément de l' hyperlien qui a lancé le téléchargement au moment où le téléchargement a été lancé.
9. Si l'opération de confiance est vraie, laissez *le nom de fichier* avoir la valeur de *nom de fichier proposé* et passez à l'étape intitulée *assainir* ci-dessous.
10. Si la ressource a un Content-Disposition en-tête `` et que cet en-tête spécifie le `attachment` type de disposition, laissez *le nom de fichier* avoir la valeur de *nom de fichier proposé*, et passez à l'étape intitulée *assainir* ci-dessous. [\[RFC6266\]](#)
11. *Aucun nom de fichier proposé* : si l'opération de confiance est vraie, ou si l'utilisateur a indiqué une préférence pour le téléchargement de la ressource en question, laissez *le nom de fichier* avoir une valeur dérivée de l' URL de la ressource d'une manière définie par l'implémentation, et passez à l'étape étiquetée *désinfecter* ci-dessous.
12. Laissez *filename* être défini sur le nom de fichier préféré de l'utilisateur ou sur un nom de fichier sélectionné par l'agent utilisateur, et passez à l'étape intitulée *aseptiser* ci-dessous.

***Si l'algorithme atteint cette étape, un téléchargement a été lancé à partir d'une origine différente de la ressource en cours de téléchargement, et l'origine n'a pas marqué le fichier comme pouvant être téléchargé, et le téléchargement n'a pas été lancé par l'utilisateur. Cela peut être dû au fait qu'un download attribut a été utilisé pour déclencher le téléchargement, ou parce que la ressource en question n'est pas d'un type pris en charge par l'agent utilisateur.***

***Cela pourrait être dangereux, car, par exemple, un serveur hostile pourrait essayer d'amener un utilisateur à télécharger sans le savoir des informations privées, puis à les télécharger à nouveau sur le serveur hostile, en faisant croire à l'utilisateur que les données proviennent du serveur hostile.***

***Ainsi, il est dans l'intérêt de l'utilisateur que l'utilisateur soit notifié d'une manière ou d'une autre que la ressource en question provient d'une***



**source assez différente, et pour éviter toute confusion, tout nom de fichier suggéré à partir de l'origine de l'interface potentiellement hostile doit être ignoré .**

13. *Sanitize* : Optionnellement, permet à l'utilisateur d'influencer *filename* . Par exemple, un agent utilisateur pourrait demander à l'utilisateur un nom de fichier, fournissant potentiellement la valeur du *nom de fichier* telle que déterminée ci-dessus comme valeur par défaut.

14. Ajustez *le nom de fichier* pour qu'il soit adapté au système de fichiers local.

Par exemple, cela peut impliquer la suppression de caractères non autorisés dans les noms de fichiers ou la suppression des espaces de début et de fin.

15. Si les conventions de la plate-forme n'utilisent en aucun cas [des extensions](#) pour déterminer les types de fichiers sur le système de fichiers, renvoyez *le nom de fichier* comme nom de fichier.

16. Soit *le type revendiqué* soit le type donné par les [métadonnées Content-Type](#) de la ressource , le cas échéant. Soit *type nommé* le type donné par [l'extension](#) du *nom de fichier* , s'il en existe un connu. Pour les besoins de cette étape, un *type* est un mappage d'un [type MIME](#) à une [extension](#) .

17. Si *le type nommé* est cohérent avec les préférences de l'utilisateur (par exemple, parce que la valeur de nom de *fichier* a été déterminée en demandant à l'utilisateur), alors renvoie *nom de fichier* comme nom de fichier.

18. Si *le type revendiqué* et *le type nommé* sont du même type (c'est-à-dire que le type donné par les [métadonnées Content-Type](#) de la ressource est cohérent avec le type donné par *filename* 's [extension](#) ), alors renvoyez *filename* comme nom de fichier.

19. Si le *type revendiqué* est connu, modifiez *le nom du fichier* pour ajouter une [extension](#) correspondant au *type revendiqué* .

Sinon, si *le type nommé* est connu pour être potentiellement dangereux (par exemple, il sera traité par les conventions de la plate-forme comme un exécutable natif, un script shell, une application HTML ou un document exécutable compatible avec les macros), modifiez éventuellement le nom de fichier pour ajouter un connu - sûr [extension](#) (par exemple " .txt").

*Cette dernière étape rendrait impossible le téléchargement d'exécutables, ce qui pourrait ne pas être souhaitable. Comme toujours, les implémenteurs sont obligés d'équilibrer la sécurité et la convivialité dans ce domaine.*

20. Renvoie *le nom de fichier* comme nom de fichier.

Aux fins de cet algorithme, une **extension** de fichier consiste en toute partie du nom de fichier que les conventions de la plate-forme dictent et qui sera utilisée pour identifier le type de fichier. Par exemple, de nombreux systèmes d'exploitation utilisent la partie du nom de fichier suivant le dernier point (" .") dans le nom de

fichier pour déterminer le type de fichier, et à partir de là, la manière dont le fichier doit être ouvert ou exécuté.

Les agents utilisateurs doivent ignorer toute information de répertoire ou de chemin fournie par la ressource elle-même, son URL et tout download attribut, pour décider où stocker le fichier résultant dans le système de fichiers de l'utilisateur.

#### 4.6.6 Audit des hyperliens

Si un lien hypertexte créé par un élément a ou area a un ping attribut, et que l'utilisateur suit le lien hypertexte, et que la valeur de l' href attribut de l'élément peut être analysée , par rapport au nœud document de l'élément , sans échec, alors l'agent utilisateur doit prendre la ping valeur de l'attribut, divisez cette chaîne sur l'espace blanc ASCII , analysez chaque jeton résultant par rapport au document de nœud de l'élément , puis exécutez ces étapes pour chaque URL de ping d'enregistrement d'URL résultant , en ignorant les jetons qui ne parviennent pas à analyser :

1. Si le schéma de l' URL ping n'est pas un schéma HTTP(S) , alors retournez.
2. En option, retour. (Par exemple, l'agent utilisateur peut souhaiter ignorer une ou toutes les URL de ping conformément aux préférences exprimées par l'utilisateur.)
3. Soit *settingsObject* l' objet de paramètres pertinent du nœud document de l'élément .
4. Soit *request* une nouvelle requête dont l'URL est ping URL , la méthode est ``POST`` , la liste d'en-tête est « (``Content-Type``, ``text/ping``) » , le corps est ``PING`` , le client est *settingsObject* , la destination est la chaîne vide, le mode d'identification est "include" , le référent est "no-referrer" , et dont l'indicateur use-URL-credentials est défini, et dont le type d'initiateur est "ping".
5. Soit *URL cible* la chaîne d'URL résultante obtenue à partir de l'analyse de la valeur de l'attribut de l'élément href, puis :

**Si l' URL de l' Document objet contenant le lien hypertexte audité et l'URL du ping ont la même origine**

**Si les origines sont différentes, mais que le schéma de l' URL du Document contenant le lien hypertexte audité n'est pas "https"**  
la requête doit inclure un Ping-From en-tête `` avec, comme valeur, l' URL du document contenant le lien hypertexte, et un Ping-To en-tête `` HTTP avec, comme valeur, l' *URL cible* .

**Sinon**

la requête doit inclure un Ping-To en-tête HTTP `` avec, comme valeur, l'URL cible . *la requête n'inclut pas d' Ping-From en-tête ``.*

6. Récupérer la requête .

Cela peut être fait en parallèle avec l'extraction primaire, et est indépendant du résultat de cette extraction.

Les agents utilisateurs doivent permettre à l'utilisateur d'ajuster ce comportement, par exemple en conjonction avec un paramètre qui désactive l'envoi d'en-têtes HTTP `` Referer (sic). Sur la base des préférences de l'utilisateur, les UA peuvent soit ignorer complètement l' ping attribut, soit ignorer sélectivement les URL de la liste (par exemple, ignorer toute URL tierce) ; ceci est explicitement pris en compte dans les étapes ci-dessus.

Les agents utilisateurs doivent ignorer tout corps d'entité renvoyé dans les réponses. Les agents utilisateurs peuvent fermer la connexion prématurément une fois qu'ils commencent à recevoir un corps de réponse.

Lorsque l' ping attribut est présent, les agents utilisateurs doivent clairement indiquer à l'utilisateur que le fait de suivre l'hyperlien entraînera également l'envoi de requêtes secondaires en arrière-plan, y compris éventuellement la liste des URL cibles réelles.

Par exemple, un agent utilisateur visuel peut inclure les noms d'hôte des URL de ping cibles avec l'URL réelle du lien hypertexte dans une barre d'état ou une info-bulle.

*L' ping attribut est redondant avec les technologies préexistantes telles que les redirections HTTP et JavaScript en permettant aux pages Web de suivre les liens hors site les plus populaires ou en permettant aux annonceurs de suivre les taux de clics.*

*Cependant, l' ping attribut offre ces avantages à l'utilisateur par rapport à ces alternatives :*

- Il permet à l'utilisateur de voir l'URL cible finale non masquée.*
- Il permet à l'UA d'informer l'utilisateur des notifications hors bande.*
- Il permet à l'utilisateur de désactiver les notifications sans perdre la fonctionnalité de lien sous-jacente.*
- Il permet à l'UA d'optimiser l'utilisation de la bande passante réseau disponible afin que la page cible se charge plus rapidement.*

*Ainsi, bien qu'il soit possible de suivre les utilisateurs sans cette fonctionnalité, les auteurs sont encouragés à utiliser l' ping attribut afin que l'agent utilisateur puisse rendre l'expérience utilisateur plus transparente.*

#### 4.6.6.1 Les en-têtes `Ping-From` et `Ping-To`

Les en-têtes de requête HTTP `Ping-From` et `Ping-To` sont inclus dans les requêtes [d'audit des liens hypertexte](#). Leur valeur est une [URL](#), [sérialisée](#).

#### 4.6.7 Types de lien



Le tableau suivant résume les types de liens définis par cette spécification, par leurs mots-clés correspondants. Ce tableau est non normatif ; les définitions réelles des types de liens sont données dans les quelques sections suivantes.

Dans cette section, le terme *document référencé* fait référence à la ressource identifiée par l'élément représentant le lien, et le terme *document courant* fait référence à la ressource dans laquelle se trouve l'élément représentant le lien.

Pour déterminer quels types de liens s'appliquent à un élément [link](#), [a](#), [area](#), ou [form](#), l'attribut `rel` de l'élément doit être [fractionné sur un espace ASCII](#). Les jetons résultants sont les mots-clés pour les types de liens qui s'appliquent à cet élément.

Sauf indication contraire, un mot-clé ne doit pas être spécifié plus d'une fois par [rel](#)attribut.

Certaines des sections qui suivent le tableau ci-dessous répertorient les synonymes de certains mots-clés. Les synonymes indiqués doivent être traités comme spécifié par les agents utilisateurs, mais ne doivent pas être utilisés dans les documents (par exemple, le mot-clé " `copyright` ").

Les mots-clés sont toujours [ASCII insensibles à la casse](#) et doivent être comparés en tant que tels.

Ainsi, `rel="next"` est identique à `rel="NEXT"`.

Les mots-clés qui sont **body-ok** affectent si [link](#) les éléments sont [autorisés dans le corps](#). Les mots-clés **body-ok** sont [dns-prefetch](#), [modulepreload](#), [pingback](#), [preconnect](#), [prefetch](#), [preload](#), [pre-render](#) et [stylesheet](#).

De nouveaux types de liens qui doivent être mis en œuvre par les navigateurs Web doivent être ajoutés à cette norme. Le reste peut être [enregistré en tant qu'extensions](#).

Type de lien	Effectuer sur...			<a href="#">corp s-ok</a>	A <a href="#">Link</a> traitement	Brève description
	<a href="#">link</a>	<a href="#">aetare a</a>	<a href="#">form</a>			
<a href="#">alternate</a>	<a href="#">Lien hypertexte</a>		<i>interdit</i>	.	.	Donne des représentations alternatives du document courant.
<a href="#">canonical</a>	<a href="#">Lien hypertexte</a>	<i>interdit</i>		.	.	Donne l'URL préférée pour le document courant.
<a href="#">author</a>	<a href="#">Lien hypertexte</a>		<i>interdit</i>	.	.	Donne un lien vers l'auteur du document ou de l'article en cours.
<a href="#">bookmark</a>	<i>interdit</i>	<a href="#">Lien hypertexte</a>	<i>interdit</i>	.	.	Donne le permalien de la section ancêtre la plus proche.
<a href="#">dns-prefetch</a>	<a href="#">Ressource externe</a>	<i>interdit</i>		Oui	.	Spécifie que l'agent utilisateur doit effectuer de manière préventive la résolution DNS pour l' <a href="#">origine</a> de la ressource cible .
<a href="#">external</a>	<i>interdit</i>	<a href="#">Annotation</a>		.	.	Indique que le document référencé ne fait pas partie du même site que le document actuel.
<a href="#">help</a>	<a href="#">Lien hypertexte</a>			.	.	Fournit un lien vers une aide contextuelle.
<a href="#">icon</a>	<a href="#">Ressource externe</a>	<i>interdit</i>		.	.	Importe une icône pour représenter le document actuel.
<a href="#">manifest</a>	<a href="#">Ressource externe</a>	<i>interdit</i>		.	.	Importe ou établit des liens vers un <a href="#">manifeste d'application</a> . <a href="#">[MANIFESTE]</a>
<a href="#">modulepreload</a>	<a href="#">Ressource externe</a>	<i>interdit</i>		Oui	.	Spécifie que l'agent utilisateur doit <a href="#">récupérer de manière préventive le script de module</a> et le stocker dans la <a href="#">carte de module</a> du document pour une évaluation ultérieure. En option, les dépendances du module

Type de lien	Effectuer sur...			<u>corp</u> <u>s-ok</u>	A ` <u>Link</u> ` traiteme nt	Brève description
	<u>link</u>	<u>aetare</u> <u>a</u>	<u>form</u>			
						peuvent également être récupérées.
<u>license</u>	<u>Lien hypertexte</u>			.	.	Indique que le contenu principal du document actuel est couvert par la licence de droit d'auteur décrite par le document référencé.
<u>next</u>	<u>Lien hypertexte</u>			.	.	Indique que le document actuel fait partie d'une série et que le document suivant de la série est le document référencé.
<u>nofollow</u>	<i>interdit</i>	<u>Annotation</u>		.	.	Indique que l'auteur ou l'éditeur original du document actuel n'approuve pas le document référencé.
<u>noopener</u>	<i>interdit</i>	<u>Annotation</u>		.	.	Crée un <u>traversable de niveau supérieur</u> avec un <u>contexte de navigation non auxiliaire</u> si le lien hypertexte en créerait autrement un qui était auxiliaire (c'est-à-dire, a une <u>target</u> valeur d'attribut appropriée).
<u>noreferrer</u>	<i>interdit</i>	<u>Annotation</u>		.	.	Aucun en-tête ` <u>Referer</u> ` (sic) ne sera inclus. De plus, a le même effet que <u>noopener</u> .
<u>opener</u>	<i>interdit</i>	<u>Annotation</u>		.	.	Crée un <u>contexte de navigation auxiliaire</u> si l'hyperlien créerait autrement un <u>traversable de niveau supérieur</u> avec un <u>contexte de navigation non auxiliaire</u> (c'est-à-dire, a " <u>_blank</u> " comme <u>target</u> valeur d'attribut).

Type de lien	Effectuer sur...			<u>corp</u> <u>s-ok</u>	A ` <u>Link</u> ` traiteme nt	Brève description
	<u>link</u>	<u>aetare</u> <u>a</u>	<u>form</u>			
<u>pingback</u>	<u>Ressour</u> <u>ce</u> <u>externe</u>	<i>interdit</i>		Oui	.	Donne l'adresse du serveur pingback qui gère les pingbacks vers le document courant.
<u>preconnect</u>	<u>Ressour</u> <u>ce</u> <u>externe</u>	<i>interdit</i>		Oui	Oui	Spécifie que l'agent utilisateur doit se connecter de manière préventive à l' <u>origine</u> de la ressource cible .
<u>prefetch</u>	<u>Ressour</u> <u>ce</u> <u>externe</u>	<i>interdit</i>		Oui	.	Spécifie que l'agent utilisateur doit <u>recupérer</u> et mettre en cache de manière préventive la ressource cible car elle est susceptible d'être requise pour une <u>navigation</u> de suivi .
<u>preload</u>	<u>Ressour</u> <u>ce</u> <u>externe</u>	<i>interdit</i>		Oui	Oui	Spécifie que l'agent utilisateur doit <u>recupérer</u> et mettre en cache de manière préventive la ressource cible pour <u>la</u> <u>navigation</u> actuelle en fonction de la <u>destination potentielle</u> donnée par l' <u>as</u> attribut (et de la <u>priorité</u> associée à la <u>destination correspond ante</u> ).
<u>prerender</u>	<u>Ressour</u> <u>ce</u> <u>externe</u>	<i>interdit</i>		Oui	.	Spécifie que l'agent utilisateur doit <u>recupérer</u> de manière préventive la ressource cible et la traiter de manière à fournir une réponse plus rapide à l'avenir.
<u>prev</u>	<u>Lien hypertexte</u>			.	.	Indique que le document actuel fait partie d'une série et que le document précédent de la série est le document référencé.

Type de lien	Effectuer sur...			<u>corp</u> <u>s-ok</u>	A ' <u>Link</u> ' traiteme nt	Brève description
	<u>link</u>	<u>area</u> <u>a</u>	<u>form</u>			
<u>search</u>	<u>Lien hypertexte</u>			.	.	Donne un lien vers une ressource qui peut être utilisée pour effectuer une recherche dans le document actuel et ses pages associées.
<u>stylesheet</u>	<u>Ressour</u> <u>ce</u> <u>externe</u>	<i>interdit</i>		Oui	.	Importe une feuille de style.
<u>tag</u>	<i>interdit</i>	<u>Lien</u> <u>hypertex</u> <u>te</u>	<i>inter</i> <i>dit</i>	.	.	Donne une balise (identifiée par l'adresse donnée) qui s'applique au document courant.

#### 4.6.7.1 Type de lien " alternate "



Le alternate mot-clé peut être utilisé avec les éléments link, area.

La signification de ce mot clé dépend des valeurs des autres attributs.

**Si l'élément est un link élément et que l' rel attribut contient également le mot-clé stylesheet**

Le alternate mot-clé modifie la signification du stylesheet mot-clé de la manière décrite pour ce mot-clé. Le alternate mot-clé ne crée pas de lien en soi.

Ici, un ensemble d' link éléments fournit des feuilles de style :

```
<!-- a persistent style sheet -->
<link rel="stylesheet" href="default.css">

<!-- the preferred alternate style sheet -->
<link rel="stylesheet" href="green.css" title="Green styles">

<!-- some alternate style sheets -->
<link rel="alternate stylesheet" href="contrast.css"
title="High contrast">
```



```
<link rel="alternate stylesheet" href="big.css" title="Big fonts">

<link rel="alternate stylesheet" href="wide.css" title="Wide screen">
```

Si le **alternate** mot clé est utilisé avec l' **type** attribut défini sur la valeur `application/rss+xml` ou la valeur `application/atom+xml`

Le mot-clé crée un [lien hypertexte](#) référant un flux de syndication (mais ne syndiquant pas nécessairement exactement le même contenu que la page actuelle).

Aux fins de la découverte automatique des flux, les agents utilisateurs doivent considérer tous [link](#) les éléments du document avec le **alternate** mot-clé utilisé et avec leur **type** attribut défini sur la valeur `application/rss+xml` ou la valeur `application/atom+xml`. Si l'agent utilisateur a le concept d'un flux de syndication par défaut, le premier élément de ce type (dans [l'ordre de l'arborescence](#) ) doit être utilisé par défaut.

Les éléments suivants [link](#) donnent des flux de syndication pour un blog :

```
<link rel="alternate" type="application/atom+xml"
href="posts.xml" title="Cool Stuff Blog">

<link rel="alternate" type="application/atom+xml"
href="posts.xml?category=robots" title="Cool Stuff Blog:
robots category">

<link rel="alternate" type="application/atom+xml"
href="comments.xml" title="Cool Stuff Blog: Comments">
```

Ces [link](#) éléments seraient utilisés par les agents utilisateurs engagés dans la découverte automatique des flux, le premier étant la valeur par défaut (le cas échéant).

L'exemple suivant propose différents flux de syndication à l'utilisateur, à l'aide [a](#) d'éléments :

```
<p>You can access the planets database using Atom feeds:</p>
<ul>
  <li><a href="recently-visited-planets.xml" rel="alternate"
type="application/atom+xml">Recently Visited Planets</a></li>
  <li><a href="known-bad-planets.xml" rel="alternate"
type="application/atom+xml">Known Bad Planets</a></li>
  <li><a href="unexplored-planets.xml" rel="alternate"
type="application/atom+xml">Unexplored Planets</a></li>
</ul>
```

Ces liens ne seraient pas utilisés dans la découverte automatique des flux.

## Sinon

Le mot-clé crée un [lien hypertexte](#) faisant référence à une autre représentation du document actuel.

La nature du document référencé est donnée par les attributs [hreflang](#), et [type](#).

Si le [alternate](#) mot-clé est utilisé avec l' [hreflang](#) attribut et que la valeur de cet attribut diffère de la [langue](#) de l' [élément de document](#) , cela indique que le document référencé est une traduction.

Si le [alternate](#) mot-clé est utilisé avec l' [type](#) attribut, il indique que le document référencé est une reformulation du document courant dans le format spécifié.

Les attributs [hreflang](#) et [type](#) peuvent être combinés lorsqu'ils sont spécifiés avec le [alternate](#) mot-clé.

L'exemple suivant montre comment vous pouvez spécifier des versions de la page qui utilisent des formats alternatifs, qui sont destinées à d'autres langues et qui sont destinées à d'autres médias :

```
<link rel=alternate href="/en/html" hreflang=en
type=text/html title="English HTML">
<link rel=alternate href="/fr/html" hreflang=fr
type=text/html title="French HTML">
<link rel=alternate href="/en/html/print" hreflang=en
type=text/html media=print title="English HTML (for
printing) ">
<link rel=alternate href="/fr/html/print" hreflang=fr
type=text/html media=print title="French HTML (for
printing) ">
<link rel=alternate href="/en/pdf" hreflang=en
type=application/pdf title="English PDF">
<link rel=alternate href="/fr/pdf" hreflang=fr
type=application/pdf title="French PDF">
```

Cette relation est transitive - c'est-à-dire que si un document est lié à deux autres documents avec le type de lien " [alternate](#) ", alors, en plus d'impliquer que ces documents sont des représentations alternatives du premier document, cela implique également que ces deux documents sont alternatifs représentations les unes des autres.

#### 4.6.7.2 Type de lien " [author](#) "

Le [author](#) mot-clé peut être utilisé avec les éléments [link](#), [a](#) et [area](#). Ce mot clé crée un [lien hypertexte](#) .

Pour les éléments [a](#) et [area](#), le [author](#) mot-clé indique que le document référencé fournit des informations supplémentaires sur l'auteur de l' [article](#) élément ancêtre

le plus proche de l'élément définissant l'hyperlien, s'il existe, ou de la page dans son ensemble, sinon.

Pour [link](#) les éléments, le [author](#) mot-clé indique que le document référencé fournit des informations supplémentaires sur l'auteur pour la page dans son ensemble.

*Le "document référencé" peut être, et est souvent, une [mailto:URL](#) donnant l'adresse e-mail de l'auteur. [\[MAILTO\]](#)*

**Synonymes** : Pour des raisons historiques, les agents utilisateurs doivent également traiter les éléments [link](#), [a](#), et qui ont un attribut avec la valeur " " comme ayant le mot-clé spécifié comme une relation de lien. [area](#) [revmade](#) [author](#)

#### 4.6.7.3 Type de lien " [bookmark](#) "

Le [bookmark](#) mot-clé peut être utilisé avec les éléments [a](#) et [area](#). Ce mot clé crée un [lien hypertexte](#) .

Le [bookmark](#) mot-clé donne un lien permanent pour l' [article](#) élément ancêtre le plus proche de l'élément de liaison en question, ou de [la section à laquelle l'élément de liaison est le plus étroitement associé](#) , s'il n'y a pas [article](#) d'éléments ancêtres.

L'extrait suivant a trois permaliens. Un agent utilisateur pourrait déterminer quel lien permanent s'applique à quelle partie de la spécification en regardant où les liens permanents sont donnés.

```
...
<body>
  <h1>Example of permalinks</h1>
  <div id="a">
    <h2>First example</h2>
    <p><a href="a.html" rel="bookmark">This permalink applies to
      only the content from the first H2 to the second H2</a>. The DIV
      isn't
        exactly that section, but it roughly corresponds to it.</p>
    </div>
    <h2>Second example</h2>
    <article id="b">
      <p><a href="b.html" rel="bookmark">This permalink applies to
        the outer ARTICLE element</a> (which could be, e.g., a blog
        post).</p>
      <article id="c">
```

```
<p><a href="c.html" rel="bookmark">This permalink applies to  
the inner ARTICLE element</a> (which could be, e.g., a blog  
comment).</p>  
</article>  
</article>  
</body>  
...
```

#### 4.6.7.4 Type de lien " **canonical** "

Le canonical mot-clé peut être utilisé avec link element. Ce mot clé crée un lien hypertexte .

Le canonical mot-clé indique que l'URL donnée par l' href attribut est l'URL préférée pour le document actuel. Cela aide les moteurs de recherche à réduire le contenu dupliqué, comme décrit plus en détail dans *The Canonical Link Relation* . [\[RFC6596\]](#)

#### 4.6.7.5 Type de lien " **dns-prefetch** "

MDN

Le dns-prefetch mot clé peut être utilisé avec link des éléments. Ce mot clé crée un lien de ressource externe . Ce mot clé est body-ok .

Le dns-prefetch mot-clé indique que l'exécution préventive de la résolution DNS pour l' origine de la ressource spécifiée est susceptible d'être bénéfique, car il est très probable que l'utilisateur aura besoin de ressources situées à cette origine , et l'expérience utilisateur serait améliorée en anticipant les coûts de latence associés avec résolution DNS. Les agents utilisateurs doivent implémenter le modèle de traitement du dns-prefetch mot-clé décrit dans *Resource Hints* . [\[CONSEILS DE RESSOURCES\]](#)

Il n'y a pas de type par défaut pour les ressources données par le dns-prefetch mot-clé.

#### 4.6.7.6 Type de lien " **external** "

Le external mot-clé peut être utilisé avec les éléments a, area et form. Ce mot-clé ne crée pas de lien hypertexte, mais annote tout autre lien hypertexte créé par l'élément (le lien hypertexte implicite, si aucun autre mot-clé n'en crée un).

Le external mot-clé indique que le lien mène à un document qui ne fait pas partie du site dont le document actuel fait partie.

#### 4.6.7.7 Type de lien " help "

Le help mot-clé peut être utilisé avec les éléments link, a, area et form. Ce mot-clé crée un lien hypertexte.

Pour les éléments a, area et form, le help mot-clé indique que le document référencé fournit des informations d'aide supplémentaires pour le parent de l'élément définissant le lien hypertexte et ses enfants.

Dans l'exemple suivant, le contrôle de formulaire est associé à une aide contextuelle. L'agent utilisateur pourrait utiliser ces informations, par exemple, afficher le document référencé si l'utilisateur appuie sur la touche "Aide" ou "F1".

```
<p><label> Topic: <input name=topic> <a href="help/topic.html"
rel="help">(Help)</a></label></p>
```

Pour link les éléments, le help mot-clé indique que le document référencé fournit une aide pour la page dans son ensemble.

Pour les éléments a et area, sur certains navigateurs, le help mot-clé fait que le lien utilise un curseur différent.

#### 4.6.7.8 Type de lien " icon "



Le icon mot-clé peut être utilisé avec link des éléments. Ce mot-clé crée un lien de ressource externe.

La ressource spécifiée est une icône représentant la page ou le site et doit être utilisée par l'agent utilisateur lors de la représentation de la page dans l'interface utilisateur.

Les icônes peuvent être des icônes auditives, des icônes visuelles ou d'autres types d'icônes. Si plusieurs icônes sont fournies, l'agent utilisateur doit sélectionner l'icône la plus appropriée en fonction des attributs type, media et sizes. S'il existe

plusieurs icônes également appropriées, les agents utilisateurs doivent utiliser la dernière déclarée dans [l'arborescence](#) au moment où l'agent utilisateur a collecté la liste des icônes. Si l'agent utilisateur essaie d'utiliser une icône mais que cette icône est déterminée, après un examen plus approfondi, comme étant en fait inappropriée (par exemple parce qu'elle utilise un format non pris en charge), alors l'agent utilisateur doit essayer l'icône suivante la plus appropriée, comme déterminé par les attributs.

*Les agents utilisateurs ne sont pas tenus de mettre à jour les icônes lorsque la liste des icônes change, mais sont encouragés à le faire.*

Il n'y a pas de type par défaut pour les ressources données par le [icon](#) mot-clé. Cependant, pour déterminer [le type de la ressource](#), les agents utilisateurs doivent s'attendre à ce que la ressource soit une image.

Les [sizes](#) mots-clés représentent la taille des icônes en pixels bruts (par opposition aux [pixels CSS](#)).

*Une icône de 50 [pixels CSS](#) de large destinée aux écrans avec une densité de pixels de périphérique de deux pixels de périphérique par [pixel CSS](#) (2x, 192 dpi) aurait une largeur de 100 pixels bruts. Cette fonctionnalité ne prend pas en charge l'indication qu'une ressource différente doit être utilisée pour les petites icônes haute résolution par rapport aux grandes icônes basse résolution (par exemple, 50 x 50 2x contre 100 x 100 1x).*

Pour analyser et traiter la valeur de l'attribut, l'agent utilisateur doit d'abord [diviser la valeur de l'attribut sur l'espace blanc ASCII](#), puis doit analyser chaque mot-clé résultant pour déterminer ce qu'il représente.

Le [any](#) mot-clé indique que la ressource contient une icône évolutive, par exemple telle que fournie par une image SVG.

Les autres mots clés doivent être analysés plus en détail comme suit pour déterminer ce qu'ils représentent :

- Si le mot-clé ne contient pas exactement un caractère U+0078 LETTRE MINUSCULE LATINE X ou U+0058 LETTRE MAJUSCULE LATINE X, alors ce mot-clé ne représente rien. Retour pour ce mot-clé.
- Laissez *largeur chaîne* être la chaîne avant le " x" ou " x".
- Laissez *la chaîne de hauteur* être la chaîne après le " x" ou " x".
- Si *la chaîne de largeur* ou *la chaîne de hauteur* commence par un caractère U+0030 DIGIT ZERO (0) ou contient des caractères autres que [des chiffres ASCII](#), alors ce mot-clé ne représente rien. Retour pour ce mot-clé.
- Appliquez les [règles d'analyse des entiers non négatifs](#) à *width string* pour obtenir *width*.

- Appliquez les [règles d'analyse des entiers non négatifs](#) à la chaîne *height* pour obtenir *height* .
- Le mot-clé indique que la ressource contient une icône bitmap avec une largeur de pixels de périphérique *de largeur* et une hauteur de pixels de périphérique *de hauteur* .

Les mots clés spécifiés sur l' [sizes](#)attribut ne doivent pas représenter des tailles d'icônes qui ne sont pas réellement disponibles dans la ressource liée.

Les [étapes de configuration de la récupération de ressource liée](#) pour ce type de ressource liée, étant donné un [link](#)élément *el* et une requête [request](#) , sont :

1. Définissez [la destination](#) de la requête sur " ".image
2. Renvoie vrai.

Les étapes du [processus d'un en-tête de lien](#) pour ce type de ressource liée ne doivent rien faire.

En l'absence de a [link](#)avec le [icon](#)mot-clé, pour [Document](#)les objets dont le [schéma](#) d' [URL](#) est un [schéma HTTP\(S\)](#) , les agents utilisateurs peuvent à la place exécuter ces étapes [en parallèle](#) :

1. Soit *request* une nouvelle [requête](#) dont l'[URL](#) est l' [enregistrement d'URL](#) obtenu en résolvant l' [URL](#) " /favicon.ico" par rapport à l' [URLDocument](#) de l'objet , le client est l' [objet de paramètres pertinent](#) de l' objet , [la destination](#) est " " , l'[indicateur synchrone](#) est défini, le [mode d'identification](#) est " " et dont l'[indicateur use-URL-credentials](#) est défini.[Document](#)`imageinclude`
2. Soit *response* le résultat de [la récupération](#) de *request* .
3. Utilisez la [réponse non sécurisée](#) de la réponse sous forme d'icône comme si elle avait été déclarée à l'aide du [icon](#)mot-clé.

L'extrait suivant montre la partie supérieure d'une application avec plusieurs icônes.

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>lsForums - Inbox</title>
    <link rel=icon href=favicon.png sizes="16x16" type="image/png">
    <link rel=icon href=windows.ico sizes="32x32 48x48"
type="image/vnd.microsoft.icon">
    <link rel=icon href=mac.icns sizes="128x128 512x512 8192x8192
32768x32768">
    <link rel=icon href=iphone.png sizes="57x57" type="image/png">
```

```

<link rel=icon href=gnome.svg sizes="any" type="image/svg+xml">
<link rel=stylesheet href=lsforums.css>
<script src=lsforums.js></script>
<meta name=application-name content="lsForums">
</head>
<body>
...

```

Pour des raisons historiques, le icon mot-clé peut être précédé du mot-clé " shortcut". Si le shortcut mot clé " " est présent, la rel valeur entière de l'attribut doit être une correspondance ASCII insensible à la casse pour la chaîne " shortcut icon" (avec un seul caractère U+0020 SPACE entre les jetons et aucun autre espace ASCII ).

#### 4.6.7.9 Type de lien " license "

Le license mot-clé peut être utilisé avec les éléments link, a, area et form. Ce mot clé crée un lien hypertexte .

Le license mot-clé indique que le document référencé fournit les termes de la licence de copyright sous lesquels le contenu principal du document actuel est fourni.

La présente spécification ne précise pas comment faire la distinction entre le contenu principal d'un document et le contenu qui n'est pas réputé faire partie de ce contenu principal. La distinction doit être claire pour l'utilisateur.

Envisagez un site de partage de photos. Une page de ce site peut décrire et afficher une photographie, et la page peut être balisée comme suit :

```

<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>Examp1 Pictures: Kissat</title>
    <link rel="stylesheet" href="/style/default">
  </head>
  <body>
    <h1>Kissat</h1>
    <nav>
      <a href="..">Return to photo index</a>
    </nav>
    <figure>
      

```



```

<figcaption>Kissat</figcaption>
</figure>
<p>One of them has six toes!</p>
<p><small><a rel="license"
href="http://www.opensource.org/licenses/mit-license.php">MIT
Licensed</a></small></p>
<footer>
  <a href="/">Home</a> | <a href="..">Photo index</a>
  <p><small>© copyright 2009 Exapl Pictures. All Rights
  Reserved.</small></p>
</footer>
</body>
</html>

```

Dans ce cas, le licenses'applique uniquement à la photo (le contenu principal du document), et non à l'ensemble du document. En particulier pas le design de la page elle-même, qui est couvert par le copyright indiqué en bas du document. Cela pourrait être rendu plus clair dans le style (par exemple, en mettant le lien de licence bien en évidence près de la photographie, tout en ayant le droit d'auteur de la page en petit texte léger au bas de la page).

**Synonymes** : Pour des raisons historiques, les agents utilisateurs doivent également traiter le mot-clé " `copyright`" comme le license mot-clé.

#### 4.6.7.10 Type de lien " `manifest`"



Le manifest mot clé peut être utilisé avec link des éléments. Ce mot clé crée un lien de ressource externe .

Le manifest mot-clé indique le fichier manifeste qui fournit les métadonnées associées au document actuel.

Il n'y a pas de type par défaut pour les ressources données par le manifest mot-clé.

Lorsqu'une application Web n'est pas installée , le moment approprié pour recupérer et traiter la ressource liée pour ce type de lien est lorsque l'agent utilisateur le juge nécessaire. Par exemple, lorsque l'utilisateur choisit d' installer l'application Web .

Pour une application Web installée , les moments appropriés pour recupérer et traiter la ressource liée pour ce type de lien sont :

- Lorsque le [lien vers la ressource externe](#) est créé sur un [link](#) élément déjà [connecté au contexte de navigation](#) .
- Lorsque l' élément du [lien vers la ressource externe devient connecté au contexte de navigation](#) .[link](#)
- Lorsque l' [href](#) attribut de l' [link](#) élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est modifié.

Dans les deux cas, seul le premier [link](#) élément de [l'arborescence](#) dont [rel](#) l'attribut contient le jeton [manifest](#) peut être utilisé.

Un agent utilisateur ne doit pas [retarder l'événement de chargement](#) pour ce type de lien.

Les [étapes de configuration de la récupération de ressource liée](#) pour ce type de ressource liée, étant donné un [link](#) élément *el* et une requête [request](#) , sont :

1. Soit *navigable* le [nœud](#) navigable du [document](#) *el* .
2. Si *navigable* est null, alors retourne false.
3. Si *navigable* n'est pas un [traversable de niveau supérieur](#) , renvoie false.
4. Définissez l' [initiateur](#) de *la requête* sur " ".[manifest](#)
5. Définissez [la destination](#) de *la requête* sur " ".[manifest](#)
6. Définissez [le mode](#) de *la requête* sur " ".[cors](#)
7. Définissez [le mode d'informations d'identification](#) de *request* sur le [mode d'informations d'identification de l'attribut CORS settings](#) pour l'attribut content de *el* .[crossorigin](#)
8. Renvoie vrai.

Pour [traiter ce type de ressource liée](#) étant donné un [link](#) élément *el* , un booléen *success* , une réponse [response](#) et une [séquence d'octets](#) *bodyBytes* :

1. Si [les métadonnées Content-Type](#) de *la réponse* ne sont pas un [type JSON MIME](#) , définissez *success* sur false.
2. Si *le succès* est vrai, [traitez le manifeste](#) donné *el* , *response* et *bodyBytes* . [\[MANIFESTE\]](#)

Les étapes du [processus d'un en-tête de lien](#) pour ce type de ressource liée ne doivent rien faire.

#### 4.6.7.11 Type de lien " `modulepreload` "



Le `modulepreload` mot clé peut être utilisé avec `link` des éléments. Ce mot clé crée un [lien de ressource externe](#) . Ce mot clé est `body-ok` .

Le `modulepreload` mot-clé est une alternative spécialisée au `preload` mot-clé, avec un modèle de traitement orienté vers le préchargement [des scripts de module](#) . En particulier, il utilise le comportement d'extraction spécifique pour les scripts de module (y compris, par exemple, une interprétation différente de l' `crossorigin` attribut), et place le résultat dans la [carte de module](#) appropriée pour une évaluation ultérieure. En revanche, un [lien de ressource externe](#) similaire utilisant le `preload` mot-clé placerait le résultat dans le cache de préchargement, sans affecter la [carte de module](#) du document .

De plus, les implémentations peuvent profiter du fait que [les scripts de module](#) déclarent leurs dépendances afin de récupérer également la dépendance du module spécifié. Ceci est conçu comme une opportunité d'optimisation, puisque l'agent utilisateur sait que, selon toute vraisemblance, ces dépendances seront également nécessaires plus tard. Il ne sera généralement pas observable sans l'utilisation de technologies telles que les techniciens de service ou la surveillance côté serveur. Notamment, les événements `load` ou appropriés `error` se produiront après la récupération du module spécifié et n'attendent aucune dépendance.

Un agent utilisateur ne doit pas [retarder l'événement de chargement](#) pour ce type de lien.

Les moments appropriés pour [récupérer et traiter la ressource liée](#) pour un tel lien sont :

- Lorsque le [lien vers la ressource externe](#) est créé sur un `link` élément déjà [connecté au contexte de navigation](#) .
- Lorsque l' élément du [lien vers la ressource externe devient connecté au contexte de navigation](#) . `link`
- Lorsque l' `href` attribut de l' `link` élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est modifié.

*Contrairement à certaines autres relations de lien, la modification des attributs pertinents (tels que `as`, `crossorigin`, et `referrerpolicy`) d'une telle relation `link` ne déclenche pas une nouvelle extraction. Cela est dû au fait que la [carte de module](#) du document a déjà été remplie par une récupération précédente, et donc une nouvelle récupération serait inutile.*

L' algorithme [de récupération et de traitement de la ressource liée](#) pour `modulepreload` les liens, étant donné un `link` élément `el` , est le suivant :

1. Si la valeur de l'attribut `elhref` est la chaîne vide, alors retournez.
2. Soit *destination* l'état actuel de l'attribut de `elas` (une [destination](#)), ou "script" s'il n'est dans aucun état.
3. Si *destination* n'est pas [de type script](#), mettez [en file d'attente une tâche d'élément](#) sur la [source de tâche réseau](#) donnée `el` pour [déclencher un événement](#) nommé `error` à `el`, et revenez.
4. [Analyser une URL](#) en fonction de la valeur de l'attribut de `elhref`, par rapport au [nœud document](#) de `el`. Si cela échoue, revenez. Sinon, laissez *url* être l'[enregistrement d'URL résultant](#).
5. Soit l'objet *settings* être l'[objet de paramètres pertinent](#) du document du [nœud](#) `el`.
6. Soit le mode d'informations d'identification le [mode d'informations d'identification de l'attribut de paramètres CORS](#) pour l'attribut de `elcrossorigin`.
7. Soit *nonce* cryptographique `el`. [\[\[CryptographicNonce\]\]](#).
8. Soit la métadonnée d'intégrité la valeur de l'attribut `elintegrity`, s'il est spécifié, ou la chaîne vide sinon.
9. Soit *referrer policy* l'état actuel de l'attribut `elreferrerpolicy`.
10. Soit *options* une [option de récupération de script](#) dont le [nonce cryptographique](#) est le *nonce cryptographique*, les [métadonnées d'intégrité](#) sont les *métadonnées d'intégrité*, les [métadonnées de l'analyseur](#) sont "not-parser-inserted", le [mode d'identification](#) est le *mode d'identification* et la [politique de référence](#) est la *politique de référence*.
11. [Récupérez un graphique de script de module de préchargement de module](#) donné *url*, *destination*, *settings object*, *options*, et avec les étapes suivantes étant donné le résultat :
  1. Si le résultat est nul, [déclenchez un événement](#) nommé `error` à `el` et retournez.
  2. [Lancer un événement](#) nommé `load` à `el`.

Les étapes du [processus d'un en-tête de lien](#) pour ce type de ressource liée ne doivent rien faire.

L'extrait de code suivant montre la partie supérieure d'une application avec plusieurs modules préchargés :

```
<!DOCTYPE html>
<html lang="en">
```

```

<title>IRCFog</title>

<link rel="modulepreload" href="app.mjs">
<link rel="modulepreload" href="helpers.mjs">
<link rel="modulepreload" href="irc.mjs">
<link rel="modulepreload" href="fog-machine.mjs">

<script type="module" src="app.mjs">
...

```

Supposons que le graphe de module pour l'application est le suivant :

Ici, nous voyons que le développeur de l'application a utilisé [modulepreload](#) pour déclarer tous les modules dans son graphe de modules, en s'assurant que l'agent utilisateur lance des récupérations pour tous. Sans un tel préchargement, l'agent utilisateur peut avoir besoin de passer par plusieurs allers-retours sur le réseau avant de découvrir `helpers.mjs`, si des technologies telles que HTTP/2 Server Push ne sont pas en jeu. De cette manière, [modulepreload link](#) les éléments peuvent être utilisés comme une sorte de "manifeste" des modules de l'application.

Le code suivant montre comment [modulepreload](#) les liens peuvent être utilisés conjointement avec [import\(\)](#) pour s'assurer que la récupération du réseau est effectuée à l'avance, de sorte que lorsqu'il [import\(\)](#) est appelé, le module est déjà prêt (mais pas évalué) dans la [carte des modules](#) :

```

<link rel="modulepreload" href="awesome-viewer.mjs">

<button onclick="import('./awesome-viewer.mjs').then(m =>
m.view())">
  View awesome thing
</button>

```

#### 4.6.7.12 Type de lien " `nofollow` "

Le `nofollow` mot-clé peut être utilisé avec les éléments `a`, `area` et `form`. Ce mot-clé ne crée pas de [lien hypertexte](#), mais [annote](#) tout autre lien hypertexte créé par l'élément (le lien hypertexte implicite, si aucun autre mot-clé n'en crée un).

Le `nofollow` mot-clé indique que le lien n'est pas approuvé par l'auteur ou l'éditeur original de la page, ou que le lien vers le document référencé a été inclus principalement en raison d'une relation commerciale entre des personnes affiliées aux deux pages.

#### 4.6.7.13 Type de lien " `noopener` "



Le `noopener` mot-clé peut être utilisé avec les éléments `a`, `area` et `form`. Ce mot-clé ne crée pas de [lien hypertexte](#), mais [annote](#) tout autre lien hypertexte créé par l'élément (le lien hypertexte implicite, si aucun autre mot-clé n'en crée un).

Le mot-clé indique que tout [traversable de niveau supérieur](#) nouvellement créé qui résulte du suivi de l' [hyperlien](#) ne contiendra pas de [contexte de navigation auxiliaire](#). Par exemple, le getter résultant `Window` renverra `opener` null.

*Voir aussi le [modèle de traitement](#).*

Cela crée généralement un [traversable de niveau supérieur](#) avec un [contexte de navigation auxiliaire](#) (en supposant qu'il n'y a pas [de navigable](#) existant dont [le nom cible](#) est " `example` ") :

```
<a href=help.html target=example>Help!</a>
```

Cela crée un [parcours de niveau supérieur](#) avec un [contexte de navigation non auxiliaire](#) (en supposant la même chose) :

```
<a href=help.html target=example rel=noopener>Help!</a>
```

Celles-ci sont équivalentes et ne naviguent que dans le [parent navigable](#) :

```
<a href=index.html target=_parent>Home</a>
<a href=index.html target=_parent rel=noopener>Home</a>
```

#### 4.6.7.14 Type de lien " `noreferrer` "



Le `noreferrer` mot-clé peut être utilisé avec les éléments `a`, `area` et `form`. Ce mot-clé ne crée pas de [lien hypertexte](#), mais [annote](#) tout autre lien hypertexte créé par l'élément (le lien hypertexte implicite, si aucun autre mot-clé n'en crée un).

Il indique qu'aucune information de référence ne doit être divulguée lorsque vous suivez le lien et implique également le `noopener` comportement du mot-clé dans les mêmes conditions.

*Voir aussi le [modèle de traitement](#) où le référent est directement manipulé.*

`<a href="..." rel="noreferrer" target="_blank"> a le même comportement que <a href="..." rel="noreferrer noopener" target="_blank">.`

#### 4.6.7.15 Type de lien " `opener` "

Le `opener` mot-clé peut être utilisé avec les éléments `a`, `area` et `form`. Ce mot-clé ne crée pas de [lien hypertexte](#), mais [annote](#) tout autre lien hypertexte créé par l'élément (le lien hypertexte implicite, si aucun autre mot-clé n'en crée un).

Le mot-clé indique que tout [traversable de niveau supérieur](#) nouvellement créé qui résulte du suivi de l' [hyperlien](#) contiendra un [contexte de navigation auxiliaire](#).

*Voir aussi le [modèle de traitement](#).*

Dans l'exemple suivant, le `opener` est utilisé pour permettre à la fenêtre contextuelle de la page d'aide de naviguer dans son ouvreuse, par exemple, dans le cas où ce que l'utilisateur recherche peut être trouvé ailleurs. Une alternative pourrait être d'utiliser une cible nommée, plutôt que `_blank`, mais cela risque d'entrer en conflit avec les noms existants.

```
<a href="..." rel=opener target=_blank>Help!</a>
```

#### 4.6.7.16 Type de lien " `pingback` "

Le `pingback` mot-clé peut être utilisé avec `link` des éléments. Ce mot-clé crée un [lien de ressource externe](#). Ce mot-clé est [body-ok](#).

Pour la sémantique du `pingback` mot-clé, voir *Pingback 1.0*. [\[PINGBACK\]](#)

#### 4.6.7.17 Type de lien " `preconnect` "

Le [preconnect](#) mot clé peut être utilisé avec [link](#) des éléments. Ce mot clé crée un [lien de ressource externe](#) . Ce mot clé est [body-ok](#) .

Le [preconnect](#) mot-clé indique que le lancement préventif d'une connexion à l' [origine](#) de la ressource spécifiée est susceptible d'être bénéfique, car il est fort probable que l'utilisateur aura besoin de ressources situées à cette [origine](#) , et l'expérience utilisateur serait améliorée en anticipant les coûts de latence associés avec l'établissement de la connexion.

Il n'y a pas de type par défaut pour les ressources données par le [preconnect](#) mot-clé.

Un agent utilisateur ne doit pas [retarder l'événement de chargement](#) pour ce type de lien.

Les moments appropriés pour [récupérer et traiter](#) ce type de lien sont :

- Lorsque le [lien vers la ressource externe](#) est créé sur un [link](#) élément déjà [connecté au contexte de navigation](#) .
- Lorsque l' élément du [lien vers la ressource externe devient connecté au contexte de navigation](#) . [link](#)
- Lorsque l' [href](#) attribut de l' [link](#) élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est modifié.
- Lorsque l' [crossorigin](#) attribut de l' [link](#) élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est défini, modifié ou supprimé.

Les étapes [de récupération et de traitement de la ressource liée](#) pour ce type de ressource liée, étant donné un [link](#) élément *e/* , consistent à [créer des options de lien](#) à partir de *e/* et à [se préconnecter](#) en fonction du résultat.

L' étape [de traitement d'un en-tête de lien](#) pour ce type de ressource liée étant donné les *options* [de traitement d'un lien](#) consiste à [préconnecter](#) les *options* données .

Pour **se préconnecter** étant donné les *options* [d'options de traitement d'un lien](#) :

1. Si [le href](#) des *options* est une chaîne vide, retourne.
2. [Analysez le href](#) des *options* par rapport à l'URL de [base des options](#) et laissez *url* être l' [enregistrement d'URL résultant](#) . Si cela échoue, revenez.
3. Si [le schéma](#) de l' *url* n'est pas un [schéma HTTP\(S\)](#) , alors retournez.
4. Soit *partitionKey* le résultat de [la détermination de la clé de partition réseau](#) en fonction de l' [environnement](#) des *options* .
5. Laissez *useCredentials* être vrai.



6. Si l' [origine croisée](#) des *options* n'est pas [Anonyme](#) et que l'[origine](#) des *options* n'a pas la [même origine](#) que l'[origine](#) de l' *url* , alors définissez *useCredentials* sur false.
7. L'agent utilisateur doit [obtenir une connexion](#) avec *partitionKey* , l'[origine](#) de l' *url* et *useCredentials* .

*Cette connexion est obtenue mais non utilisée directement. Il restera dans le [pool de connexion](#) pour une utilisation ultérieure.*

L'agent utilisateur doit tenter d'initier une préconnexion et d'effectuer la poignée de main de connexion complète (DNS+TCP pour HTTP et DNS+TCP+TLS pour les origines HTTPS) chaque fois que possible, mais est autorisé à choisir d'effectuer une poignée de main partielle (DNS uniquement pour HTTP , et DNS ou DNS+TCP pour les origines HTTPS), ou ignorez-le entièrement, en raison de contraintes de ressources ou pour d'autres raisons.

Le nombre optimal de connexions par origine dépend du protocole négocié, du profil de connectivité actuel des utilisateurs, des ressources de périphérique disponibles, des limites de connexion globales et d'autres variables spécifiques au contexte. En conséquence, la décision du nombre de connexions à ouvrir est reportée à l'agent utilisateur.

#### 4.6.7.18 Type de lien " [prefetch](#) "

##### MDN

Le [prefetch](#) mot clé peut être utilisé avec [link](#) des éléments. Ce mot clé crée un [lien de ressource externe](#) . Ce mot clé est [body-ok](#) .

Le [prefetch](#) mot-clé indique que [la récupération](#) et la mise en cache de manière préventive de la ressource spécifiée ou du document du même site sont susceptibles d'être bénéfiques, car il est fort probable que l'utilisateur aura besoin de cette ressource pour de futures navigations.

Il n'y a pas de type par défaut pour les ressources données par le [prefetch](#) mot-clé.

L' algorithme [de récupération et de traitement de la ressource liée](#) pour [prefetch](#) les liens, étant donné un [link](#) élément *e/* , est le suivant :

1. Si la valeur de l'attribut *e/*[href](#) est la chaîne vide, alors retournez.
2. Soit *options* le résultat de [la création d'options de lien](#) à partir de *e/* .
3. Définissez [la destination](#) des *options* sur la chaîne vide.

4. Soit *request* le résultat de [la création d'une requête de lien](#) avec *options* .
5. Si *la demande* est nulle, alors retour.
6. Définissez l' [initiateur](#) de *la requête* sur " ".prefetch
7. Soit *processPrefetchResponse* les étapes suivantes étant donné une réponse [réponse](#) et null, un échec ou une [séquence d'octets](#) *bytesOrNull* :
  1. Si *la réponse* est une [erreur réseau](#) , [déclenchez un événement](#) nommé [error](#) à *el* .
  2. Sinon, [déclenchez un événement](#) nommé [load](#) à *el* .
8. L'agent utilisateur doit [récupérer](#) *la requête* , avec [processResponseConsumeBody](#) défini sur *processPrefetchResponse* . Les agents utilisateurs peuvent retarder la récupération de *la requête* pour donner la priorité à d'autres requêtes qui sont nécessaires pour le document en cours.

Les étapes du [processus d'un en-tête de lien](#) pour ce type de ressource liée ne doivent rien faire.

#### 4.6.7.19 Type de lien " [preload](#) "



Le [preload](#) mot clé peut être utilisé avec [link](#) des éléments. Ce mot clé crée un [lien de ressource externe](#) . Ce mot clé est [body-ok](#) .

Le [preload](#) mot-clé indique que l'agent utilisateur récupérera [et](#) mettra en cache de manière préventive la ressource spécifiée en fonction de la [destination potentielle](#) donnée par l' [as](#) attribut (et de la [priorité](#) associée à la [destination correspondante](#) ), car il est fort probable que l'utilisateur aura besoin de cette ressource pour le courant la navigation. [\[PRÉCHARGER\]](#)

*Les agents utilisateurs peuvent effectuer des opérations supplémentaires lorsqu'une ressource est chargée, comme [le décodage préventif d'images](#) ou [la création de feuilles de style](#) . Cependant, ces opérations supplémentaires ne peuvent pas avoir d'effets observables.*

Il n'y a pas de type par défaut pour les ressources données par le [preload](#) mot-clé.

Un agent utilisateur ne doit pas [retarder l'événement de chargement](#) pour ce type de lien.

Les moments appropriés pour [récupérer et traiter la ressource liée](#) pour un tel lien sont :

- Lorsque le [lien vers la ressource externe](#) est créé sur un [link](#) élément déjà [connecté au contexte de navigation](#) .
- Lorsque l' élément du [lien vers la ressource externe devient connecté au contexte de navigation](#) .[link](#)
- Lorsque l' [href](#) attribut de l' [link](#) élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est modifié.
- Lorsque l' [as](#) attribut de l' [link](#) élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est modifié.
- Lorsque l' [type](#) attribut de l' [link](#) élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) , mais qui n'a pas été obtenu auparavant en raison de l' [type](#) attribut spécifiant un type non pris en charge pour la [destination](#) de la demande , est défini, supprimé ou modifié.
- Lorsque l' [media](#) attribut de l' [link](#) élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) , mais qui n'a pas été obtenu auparavant car l' [media](#) attribut ne [correspond pas à l'environnement](#) , est modifié ou supprimé.

A [Document](#) a une **carte de ressources préchargées** , qui est une [carte](#) , initialement vide.

Une **clé de préchargement** est une [structure](#) . Il comporte les [éléments](#) suivants :

#### **URL**

Une [URL](#)

#### **destination**

Un string

#### **mode**

Un [mode de requête](#) , soit " same-origin", " cors" ou " no-cors"

#### **mode d'identification**

Un [mode identifiants](#)

Une **entrée de préchargement** est une [structure](#) . Il comporte les [éléments](#) suivants :

#### **métadonnées d'intégrité**

Un string

#### **réponse**

Null ou une [réponse](#)

### **sur réponse disponible**

Null, ou un algorithme acceptant une [réponse](#) ou null

Pour **consommer une ressource préchargée** pour [Window](#) window , étant donné une [URL](#) url , une chaîne destination , une chaîne mode , une chaîne authenticationMode , une chaîne integrityMetadata et onResponseAvailable , qui est un algorithme acceptant une [réponse](#) :

1. Soit key une [clé de préchargement](#) dont [l'URL](#) est url , [la destination](#) est la destination , [le mode](#) est le mode et [le mode des informations d'identification](#) est le mode d'informations d'identification .
2. Soit *préchargements* la carte des [ressources préchargées associées](#) à la fenêtre .[Document](#)
3. Si la clé n'existe pas [dans](#) preloads , renvoie false.
4. Laisser l'entrée être *précharges* [ clé ].
5. Soit *consumerIntegrityMetadata* le résultat de [l'analyse](#) de integrityMetadata .
6. Soit *preloadIntegrityMetadata* le résultat de [l'analyse des métadonnées d'intégrité](#) de l'entrée .
7. Si aucune des conditions suivantes ne s'applique :
  - *consumerIntegrityMetadata* est no metadata;
  - *consumerIntegrityMetadata* est égal à *preloadIntegrityMetadata* ; ou

Cette comparaison ignorerait les options d'intégrité inconnues. Voir [numéro 116](#).

8. puis renvoie faux.

9. Une non-concordance des métadonnées d'intégrité entre le préchargement et le consommateur, même si les deux correspondent aux données, conduirait à une extraction supplémentaire du réseau.

10. Il est important que [les erreurs réseau](#) soient ajoutées au cache de préchargement afin que si une demande de préchargement entraîne une erreur, la réponse erronée ne soit pas redemandée ultérieurement au réseau. Cela a également des implications sur la sécurité ; considérez le cas où un développeur spécifie des métadonnées d'intégrité de sous-ressource sur une demande de préchargement, mais pas la demande de ressource suivante. Si la demande de préchargement échoue à la vérification de l'intégrité de la sous-ressource et est rejetée, la demande de ressource récupérera et consommera une réponse potentiellement malveillante du réseau sans vérifier son intégrité. [\[ISR\]](#)

11. [Supprimer](#) les précharges [ touche ].

12. Si [la réponse](#) de l'entrée est nulle, définissez la réponse de l'entrée [disponible](#) sur `onResponseAvailable` .
13. Sinon, appelez `onResponseAvailable` avec [la réponse](#) de l' entrée .
14. Renvoie vrai.

Pour les besoins de cette section, un *type de chaîne* **correspond** à une *destination* de chaîne si l'algorithme suivant renvoie true :

1. Si *type* est une chaîne vide, alors retourne vrai.
2. Si *destination* est " `fetch`", alors retourne vrai.
3. Soit *mimeTypeRecord* le résultat de [l'analyse](#) *type* .
4. Si *mimeTypeRecord* est un échec, renvoie false.
5. Si *mimeTypeRecord* n'est pas [pris en charge par l'agent utilisateur](#) , renvoie false.
6. Renvoie vrai si l'une des conditions suivantes est vraie :
  - *destination* est " `audio`" ou " `video`", et *mimeTypeRecord* est un [type MIME audio ou vidéo](#)
  - *destination* est une [destination de type script](#) et *mimeTypeRecord* est un [type JavaScript MIME](#)
  - *destination* est " `image`" et *mimeTypeRecord* est un [type MIME d'image](#)
  - *destination* est " `font`" et *mimeTypeRecord* est un [type de police MIME](#)
  - *destination* est " `style`" et [l'essence](#) de *mimeTypeRecord* est [text/css](#)
  - *destination* est " `track`" et [l'essence](#) de *mimeTypeRecord* est [text/vtt](#)
7. Renvoie faux.

Pour **créer une clé de préchargement** pour une *demande* [de requête](#) , renvoyez une nouvelle [clé de préchargement](#) dont [l'URL](#) est l' [URL](#) de la *requête* , [la destination](#) est [la destination](#) de la *requête* , [le mode](#) est [le mode](#) de la *requête* et [le mode d'informations d'identification](#) est [le mode d'informations d'identification](#) de la *requête* .

Pour **précharger** un [lien](#) , les *options* de traitement des options et un *processResponse* facultatif , qui est un algorithme acceptant une [réponse](#) :

1. Si [le type](#) des *options* ne [correspond pas à la destination](#) des *options* , alors retournez.
2. Si [la destination](#) des *options* est " " et que [l' ensemble source](#) des *options* n'est pas nul , alors définissez [le href](#) des *options* sur le résultat de [la sélection d' une source d' image](#) à partir de l' [ensemble source](#) des *options* .`image`

3. Soit *request* le résultat de [la création d'une requête de lien](#) avec *options* .
4. Si *la demande* est nulle, alors retour.
5. Soit *unsafeEndTime* égal à 0.
6. Soit *entry* une nouvelle [entrée de préchargement](#) dont [les métadonnées d'intégrité](#) sont [l'intégrité](#) des *options* .
7. Soit *key* le résultat de [la création d'une clé de préchargement](#) donnée *request* .
8. Si [le document](#) des *options* est " ", alors définissez [le type d'initiateur](#) de *la requête* sur " ".`pendingearly hint`
9. Laissez *le contrôleur* être nul.
10. Soit *reportTiming* étant donné un [Document](#) *document* pour [rapporter la synchronisation](#) pour *le contrôleur* donné [l'objet global pertinent](#) du *document* .
11. Définissez *controller* sur le résultat de [l'extraction](#) de *request* , avec [processResponseConsumeBody](#) défini sur les étapes suivantes étant donné une *réponse* [de réponse](#) et null, un échec ou une [séquence d'octets](#) *bodyBytes* :

1. Si *bodyBytes* est une [séquence d'octets](#) , alors définissez [le corps](#) de *la réponse* sur *bodyBytes* [en tant que corps](#) .

*En utilisant [processResponseConsumeBody](#) , nous avons [extrait le corps](#) entier . Cela est nécessaire pour garantir que le préchargeur charge tout le corps à partir du réseau, que la précharge soit consommée ou non (ce qui est incertain à ce stade). Cette étape réinitialise ensuite le corps de la requête dans un nouveau corps contenant les mêmes octets, afin que d'autres spécifications puissent le lire au moment de la consommation réelle, bien que nous l'ayons déjà fait une fois.*

2. Sinon, définissez *la réponse* à une [erreur réseau](#) .
3. Définissez *unsafeEndTime* sur l' [heure actuelle partagée non sécurisée](#) .
4. Si [le document](#) des *options* n'est pas nul, appelez *reportTiming* étant donné [le document](#) des *options* .
5. Si *entry*'s [on response available](#) est null, alors définissez [la réponse](#) de *entry* sur *response* ; sinon appelez l' *entrée* [sur réponse disponible](#) donnée *réponse* .
6. Si *processResponse* est donné, appelez *processResponse* avec *response* .

12. Soit *commit* les étapes suivantes étant donné un Document *document* :

1. If *entry*'s response is not null, then call *reportTiming* given *document*.
  2. Set *document*'s map of preloaded resources[*key*] to *entry*.
13. If *options*'s document is null, then set *options*'s on document ready to *commit*. Otherwise call *commit* with *options*'s document.

The fetch and process the linked resource steps for this type of linked resource, given a link element *el*, are:

1. Mettez à jour le jeu de sources pour *el*.
2. Soit *options* le résultat de la création d'options de lien à partir de *el*.
3. *Options* de préchargement, avec les étapes suivantes avec une réponse :
  1. Si *la réponse* est une erreur réseau, déclenchez un événement nommé error à *el*. Sinon, déclenchez un événement nommé load à *el*.

Le comportement réel des navigateurs est différent de la spécification ici, et la possibilité de modifier le comportement n'a pas encore été étudiée. Voir numéro 1142.

Le processus d'une étape d'en-tête de lien pour ce type de lien étant donné les *options* de traitement d'un lien consiste à précharger les *options*.

#### 4.6.7.20 Type de lien " prerender "



Le prerender mot clé peut être utilisé avec link des éléments. Ce mot clé crée un lien de ressource externe. Ce mot clé est body-ok.

Le prerender mot-clé indique que la ressource spécifiée pourrait être requise par la prochaine navigation, et il peut donc être avantageux non seulement de recupérer la ressource de manière préventive, mais aussi de la traiter, par exemple en recupérant ses sous-ressources ou en effectuant un rendu. Les agents utilisateurs doivent implémenter le modèle de traitement du prerender mot-clé décrit dans *Resource Hints*. [CONSEILS DE RESSOURCES]

Il n'y a pas de type par défaut pour les ressources données par le prerender mot-clé.



#### 4.6.7.21 Type de lien " **search** "

Le **search** mot-clé peut être utilisé avec les éléments **link**, **a**, **area** et **form**. Ce mot clé crée un **lien hypertexte** .

Le **search** mot-clé indique que le document référencé fournit une interface spécifiquement pour rechercher le document et ses ressources associées.

*Les documents de description OpenSearch peuvent être utilisés avec **link** des éléments et le **search** type de lien pour permettre aux agents utilisateurs de découvrir automatiquement les interfaces de recherche. [\[OUVRIR RECHERCHE\]](#)*

#### 4.6.7.22 Type de lien " **stylesheet** "

Le **stylesheet** mot clé peut être utilisé avec **link** des éléments. Ce mot clé crée un **lien de ressource externe** qui contribue au modèle de traitement de style. Ce mot clé est **body-ok** .

La ressource spécifiée est une **feuille de style CSS** qui décrit comment présenter le document.



Si le **alternate** mot-clé est également spécifié sur l' **link** élément, alors **le lien est une feuille de style alternative** ; dans ce cas, l' **title** attribut doit être spécifié sur l' **link** élément, avec une valeur non vide.

Le type par défaut des ressources donné par le **stylesheet** mot-clé est **text/css**.

Un **link** élément de ce type est **implicitement potentiellement bloquant le rendu** si l'élément a été créé par l'analyseur de son **nœud document** .

Les moments appropriés pour **récupérer et traiter** ce type de lien sont :

- Lorsque le **lien vers la ressource externe** est créé sur un **link** élément déjà **connecté au contexte de navigation** .
- Lorsque l' élément du **lien vers la ressource externe devient connecté au contexte de navigation** . **link**
- Lorsque l' **href** attribut de l' **link** élément d'un **lien de ressource externe** qui est déjà **connecté au contexte de navigation** est modifié.



- Lorsque l' `disabled` attribut de l' `link` élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est défini, modifié ou supprimé.
- Lorsque l' `crossorigin` attribut de l' `link` élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est défini, modifié ou supprimé.
- Lorsque l' `type` attribut de l' `link` élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) est défini ou remplacé par une valeur qui ne correspond pas ou plus aux [métadonnées Content-Type](#) de la ressource externe précédemment obtenue, le cas échéant.
- Lorsque l' `type` attribut de l' `link` élément d'un [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) , mais qui n'a pas été obtenu auparavant en raison de l' `type` attribut spécifiant un type non pris en charge, est supprimé ou modifié.
- Lorsque le [lien de ressource externe](#) qui est déjà [connecté au contexte de navigation](#) passe d' [une feuille de style alternative](#) à n'en étant pas une, ou vice versa.

**Quirk** : si le document a été défini en [mode Quirks](#) , a la [même origine](#) que l' [URL](#) de la ressource externe et que les [métadonnées Content-Type](#) de la ressource externe ne sont pas un type de feuille de style pris en charge, l'agent utilisateur doit à la place le supposer pour être `text/css`.

Les [étapes de configuration de la récupération de ressource liée](#) pour ce type de ressource liée, étant donné un `link` élément *el* et une requête `request` , sont :

1. Si l'attribut de `el` `disabled` est défini, alors renvoie false.
2. Si `el` [contribue à une feuille de style de blocage de script](#) , [ajoutez](#) `el` au [jeu de feuilles de style de blocage de script](#) de son [document de nœud](#) .
3. Si la valeur de l'attribut `el` [correspond à l'environnement](#) et que `el` est [potentiellement bloquant le rendu](#) , alors [bloquez le rendu](#) sur `el` `media` .
4. Si `el` est actuellement [render-blocking](#) , alors définissez [le render-blocking](#) de `request` sur true.
5. Renvoie vrai.

Voir [le problème #968](#) pour les plans d'utilisation de l' algorithme [de récupération CSSOM d'une feuille de style CSS](#) au lieu de l' extraction par défaut et du traitement de l' algorithme [de ressource liée](#) . Dans l'intervalle, toute [demande de sous-ressource critique](#) doit avoir son [paramètre de blocage de rendu](#) défini sur le fait que l' élément soit ou non en train de [bloquer le rendu](#) `.link`

Pour [traiter ce type de ressource liée](#) étant donné un [link](#)élément *e/* , un booléen *success* , une réponse [response](#) et une [séquence d'octets](#) *bodyBytes* :

1. Si les [métadonnées Content-Type](#) de la ressource ne sont pas [text/css](#), définissez *success* sur false.
2. Si *e/* ne crée plus de [lien de ressource externe](#) qui contribue au modèle de traitement de style, ou si, puisque la ressource en question a été [récupérée](#) , il est devenu approprié de la [récupérer](#) à nouveau, alors :
  1. [Supprimer](#) *e/* du [jeu de feuilles de style de blocage de script](#) du document [de nœud](#) *e/* .
  2. Retour.
3. Si *e/* a une [feuille de style CSS associée](#) , [supprimez la feuille de style CSS](#) .
4. Si le succès est vrai, alors :
  1. [Créez une feuille de style CSS](#) avec les propriétés suivantes :

#### [taper](#)

[text/css](#)

#### [emplacement](#)

La [chaîne d'URL résultante](#) déterminée lors de l' [extraction et du traitement de l'](#) algorithme de ressource liée.

*C'est avant que les redirections ne soient appliquées.*

#### [nœud propriétaire](#)

*élément*

#### [médias](#)

L' [media](#)attribut de l'*élément* .

*Il s'agit d'une référence à l'attribut (éventuellement absent à ce moment), plutôt qu'une copie de la valeur actuelle de l'attribut. CSSOM définit ce qui se passe lorsque l'attribut est défini, modifié ou supprimé de manière dynamique.*

#### [titre](#)

L' [title](#)attribut de *element* , si *element* est [dans une arborescence de documents](#) , ou la chaîne vide sinon.

*Il s'agit de la même manière d'une référence à l'attribut, plutôt que d'une copie de la valeur actuelle de l'attribut.*

#### [drapeau alternatif](#)

Défini si [le lien est une feuille de style alternative](#) et si l' *élément* [explicitement activé](#) est faux ; sinon.

### drapeau d'origine propre

Défini si la ressource est [CORS-same-origin](#) ; sinon.

### feuille de style CSS mère règle CSS du propriétaire

nul

### drapeau désactivé

Laissé à sa valeur par défaut.

### Règles CSS

Laissé non initialisé.

Cela ne semble pas juste. Nous devrions probablement utiliser *bodyBytes* ? Suivi en tant que [numéro 2997](#) .

[L'encodage de l'environnement](#) CSS est le résultat de l'exécution des étapes suivantes : [\[CSSSYNTAX\]](#)

1. Si l'élément a un [charset](#) attribut, [obtenez un encodage](#) à partir de la valeur de cet attribut. Si cela réussit, retournez l'encodage résultant. [\[CODAGE\]](#)
2. Sinon, renvoie l' [encodage des caractères du document](#) . [\[DOM\]](#)
2. [Lancer un événement](#) nommé [load](#) à *e/* .
5. Sinon, [déclenchez un événement](#) nommé [error](#) à *e/* .
6. Si *e/* [contribue à une feuille de style bloquant les scripts](#) , alors :
  1. [Assert](#) : le [jeu de feuilles de style de blocage de script](#) du [document](#) du nœud *e/* [contient](#) *e/* .
  2. [Supprimez](#) *e/* de [l'ensemble de feuilles de style de blocage de script](#) de son [document de nœud](#) .
7. [Débloquer le rendu](#) sur *e/* .

Les étapes du [processus d'un en-tête de lien](#) pour ce type de ressource liée ne doivent rien faire.

#### 4.6.7.23 Type de lien " [tag](#) "

Le tag mot-clé peut être utilisé avec les éléments a et area. Ce mot clé crée un lien hypertexte .

Le tag mot clé indique que la *balise* représentée par le document référencé s'applique au document actuel.

*Puisqu'il indique que la balise s'applique au document actuel , il serait inapproprié d'utiliser ce mot-clé dans le balisage d'un nuage de balises , qui répertorie les balises populaires sur un ensemble de pages.*

Ce document concerne certaines gemmes, et il est donc *étiqueté* avec " <https://en.wikipedia.org/wiki/Gemstone> " pour le catégoriser sans ambiguïté comme s'appliquant au type de gemmes "bijou", et non, par exemple, aux villes des États-Unis, au format de package Ruby ou à la Suisse. classe de locomotive :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>My Precious</title>
  </head>
  <body>
    <header><h1>My precious</h1> <p>Summer 2012</p></header>
    <p>Recently I managed to dispose of a red gem that had been
    bothering me. I now have a much nicer blue sapphire.</p>
    <p>The red gem had been found in a bauxite stone while I was
    digging
    out the office level, but nobody was willing to haul it away. The
    same red gem stayed there for literally years.</p>
    <footer>
      Tags: <a rel=tag
href="https://en.wikipedia.org/wiki/Gemstone">Gemstone</a>
    </footer>
  </body>
</html>
```

Dans ce document, il y a deux articles. Le tag lien " ", cependant, s'applique à toute la page (et le ferait où qu'il soit placé, y compris s'il se trouvait dans les article éléments).

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>Gem 4/4</title>
  </head>
  <body>
    <article>
```

```

<h1>801: Steinbock</h1>

<p>The number 801 Gem 4/4 electro-diesel has an ibex and was
rebuilt in 2002.</p>

</article>

<article>

<h1>802: Murmeltier</h1>

<figure>

  <figcaption>The 802 in the 1980s, above Lago
Bianco.</figcaption>

</figure>

<p>The number 802 Gem 4/4 electro-diesel has a marmot and was
rebuilt in 2003.</p>

</article>

<p class="topic"><a rel=tag
href="https://en.wikipedia.org/wiki/Rhaetian_Railway_Gem_4/4">Gem
4/4</a></p>

</body>

</html>

```

#### 4.6.7.24 Types de liens séquentiels

Certains documents font partie d'une séquence de documents.

Une séquence de documents est une séquence où chaque document peut avoir un *frère précédent* et un *frère suivant*. Un document sans frère précédent est le début de sa séquence, un document sans frère suivant est la fin de sa séquence.

Un document peut faire partie de plusieurs séquences.

##### 4.6.7.24.1 Type de lien " **next** "

Le next mot-clé peut être utilisé avec les éléments link, a, area et form. Ce mot clé crée un lien hypertexte.

Le next mot-clé indique que le document fait partie d'une séquence et que le lien mène au document qui est le document logique suivant dans la séquence.

Lorsque le next mot-clé est utilisé avec un link élément, les agents utilisateurs doivent implémenter l'un des modèles de traitement décrits dans *Resource Hints* , c'est-à-dire qu'ils doivent traiter ces liens comme s'ils utilisaient l'un des mots-clés dns-prefetch, preconnect, prefetch ou prerender . L'indication de ressource que l'agent utilisateur souhaite utiliser dépend de l'implémentation ; par exemple, un agent utilisateur peut souhaiter utiliser preconnect l'indice le moins coûteux lorsqu'il essaie de conserver les données, la puissance de la batterie ou la puissance de traitement, ou peut souhaiter choisir un indice de ressource en fonction de l'analyse heuristique du comportement passé de l'utilisateur dans des scénarios similaires. [\[CONSEILS DE RESSOURCES\]](#)

#### 4.6.7.24.2 Type de lien "prev"

Le prev mot-clé peut être utilisé avec les éléments link, a, area et form. Ce mot clé crée un [lien hypertexte](#) .

Le prev mot-clé indique que le document fait partie d'une séquence et que le lien mène au document qui est le document logique précédent dans la séquence.

**Synonymes** : Pour des raisons historiques, les agents utilisateurs doivent également traiter le mot-clé "previous" comme le prev mot-clé.

#### 4.6.7.25 Autres types de liens

**Les extensions de l'ensemble prédéfini de types de liens** peuvent être enregistrées sur la [page des microformats pour les valeurs rel existantes](#) . [\[MFREL\]](#)

N'importe qui est libre d'éditer la page des microformats pour les valeurs rel existantes à tout moment pour ajouter un type. Les types d'extension doivent être spécifiés avec les informations suivantes :

##### **Mot-clé**

La valeur réelle en cours de définition. La valeur ne doit pas prêter à confusion avec toute autre valeur définie (par exemple, ne différant que par la casse).

Si la valeur contient un caractère U+003A COLON (:), il doit également s'agir d'une [URL absolue](#) .

##### **Effectuer sur...link**

L'un des éléments suivants :

##### **Interdit**

Le mot-clé ne doit pas être spécifié sur link les éléments.

**Lien hypertexte**

Le mot-clé peut être spécifié sur un [link](#)élément ; il crée un [lien hypertexte](#) .

**Ressource externe**

Le mot-clé peut être spécifié sur un [link](#)élément ; il crée un [lien de ressource externe](#) .

**Effet sur... [a](#)[et](#)[area](#)**

L'un des éléments suivants :

**Interdit**

Le mot-clé ne doit pas être spécifié sur les éléments [a](#)[et](#)[area](#).

**Lien hypertexte**

Le mot-clé peut être spécifié sur les éléments [a](#)[et](#)[area](#); il crée un [lien hypertexte](#) .

**Ressource externe**

Le mot-clé peut être spécifié sur les éléments [a](#)[et](#)[area](#); il crée un [lien de ressource externe](#) .

**Annotation de lien hypertexte**

Le mot-clé peut être spécifié sur les éléments [a](#)[et](#)[area](#); il [annote](#) les autres [hyperliens](#) créés par l'élément.

**Effectuer sur... [form](#)**

L'un des éléments suivants :

**Interdit**

Le mot-clé ne doit pas être spécifié sur [form](#)les éléments.

**Lien hypertexte**

Le mot-clé peut être spécifié sur [form](#)des éléments ; il crée un [lien hypertexte](#) .

**Ressource externe**

Le mot-clé peut être spécifié sur [form](#)des éléments ; il crée un [lien de ressource externe](#) .

**Annotation de lien hypertexte**

Le mot-clé peut être spécifié sur [form](#)des éléments ; il [annote](#) les autres [hyperliens](#) créés par l'élément.

**Brève description**

Une brève description non normative de la signification du mot-clé.

**spécification**

Un lien vers une description plus détaillée de la sémantique et des exigences du mot-clé. Il peut s'agir d'une autre page du wiki ou d'un lien vers une page externe.

**Synonymes**

Une liste d'autres valeurs de mots clés qui ont exactement les mêmes exigences de traitement. Les auteurs ne doivent pas utiliser les valeurs

définies comme synonymes, elles sont uniquement destinées à permettre aux agents utilisateurs de prendre en charge le contenu hérité. N'importe qui peut supprimer les synonymes qui ne sont pas utilisés dans la pratique ; seuls les noms qui doivent être traités comme des synonymes pour la compatibilité avec le contenu hérité doivent être enregistrés de cette manière.

## **Statut**

L'un des éléments suivants :

### **Proposé**

Le mot clé n'a pas fait l'objet d'un examen approfondi ni d'une approbation par les pairs. Quelqu'un l'a proposé et l'utilise ou l'utilisera bientôt.

### **Ratifié**

Le mot-clé a reçu un large examen et approbation par les pairs. Il a une spécification qui définit sans ambiguïté comment gérer les pages qui utilisent le mot-clé, y compris lorsqu'elles l'utilisent de manière incorrecte.

### **Discontinué**

Le mot clé a fait l'objet d'un large examen par les pairs et il a été jugé insuffisant. Les pages existantes utilisent ce mot-clé, mais les nouvelles pages doivent l'éviter. Les entrées "brève description" et "spécification" donneront des détails sur ce que les auteurs devraient utiliser à la place, le cas échéant.

Si un mot-clé s'avère redondant avec des valeurs existantes, il doit être supprimé et répertorié comme synonyme de la valeur existante.

Si un mot-clé est enregistré dans l'état "proposé" pendant une période d'un mois ou plus sans être utilisé ou spécifié, il peut alors être supprimé du registre.

Si un mot-clé est ajouté avec le statut "proposé" et s'avère redondant avec les valeurs existantes, il doit être supprimé et répertorié comme synonyme de la valeur existante. Si un mot clé est ajouté avec le statut "proposé" et s'avère dangereux, il doit être remplacé par le statut "abandonné".

N'importe qui peut modifier le statut à tout moment, mais ne doit le faire que conformément aux définitions ci-dessus.

Les vérificateurs de conformité doivent utiliser les informations données sur la page des microformats pour les valeurs rel existantes afin d'établir si une valeur est autorisée ou non : les valeurs définies dans cette spécification ou marquées comme "proposées" ou "ratifiées" doivent être acceptées lorsqu'elles sont utilisées sur les éléments pour lesquels ils s'appliquent comme décrit dans le champ "Effet sur...", tandis que les valeurs marquées comme "abandonnées" ou non répertoriées dans cette spécification ou sur la page susmentionnée doivent être rejetées comme non valides. Les contrôleurs de conformité peuvent mettre ces informations en cache (par exemple pour des raisons de performances ou pour éviter l'utilisation d'une connectivité réseau non fiable).



Lorsqu'un auteur utilise un nouveau type non défini ni par cette spécification ni par la page wiki, les vérificateurs de conformité doivent proposer d'ajouter la valeur au wiki, avec les détails décrits ci-dessus, avec le statut "proposé".

Les types définis comme extensions dans la [page des microformats pour les valeurs rel existantes](#) avec le statut "proposé" ou "ratifié" peuvent être utilisés avec l' `rel` attribut sur les éléments [link](#), [a](#) et [area](#) conformément au champ "Effet sur...". [\[MFREL\]](#)

## 4.7 Modifications

Les éléments [ins](#) et [del](#) représentent les modifications apportées au document.

### 4.7.1 L' `ins` élément



#### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

#### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

#### Modèle de contenu :

[Transparente](#) .

#### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

#### Attributs de contenu :

[Attributs globaux](#)  
[cite](#)— Lien vers la source de la citation ou plus d'informations sur la modification  
[datetime](#)— Date et (facultativement) heure du changement

#### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

#### Interface DOM :

Utilisations [HTMLModElement](#).

L' insélément représente un ajout au document.

Ce qui suit représente l'ajout d'un seul paragraphe :

```
<aside>
  <ins>
    <p> I like fruit. </p>
  </ins>
</aside>
```

De même que ce qui suit, car tout dans l' asideélément ici compte comme contenu de phrasé et il n'y a donc qu'un seul paragraphe :

```
<aside>
  <ins>
    Apples are <em>tasty</em>.
  </ins>
  <ins>
    So are pears.
  </ins>
</aside>
```

insles éléments ne doivent pas dépasser les limites implicites des paragraphes .

L'exemple suivant représente l'ajout de deux paragraphes dont le second a été inséré en deux parties. Le premier insélément de cet exemple franchit donc une limite de paragraphe, qui est considérée comme de mauvaise forme.

```
<aside>
  <!-- don't do this -->
  <ins datetime="2005-03-16 00:00Z">
    <p> I like fruit. </p>
    Apples are <em>tasty</em>.
  </ins>
  <ins datetime="2007-12-19 00:00Z">
    So are pears.
  </ins>
</aside>
```

Voici une meilleure façon de marquer cela. Il utilise plus d'éléments, mais aucun des éléments ne dépasse les limites de paragraphe implicites.

```
<aside>
  <ins datetime="2005-03-16 00:00Z">
```

```
<p> I like fruit. </p>
</ins>
<ins datetime="2005-03-16 00:00Z">
  Apples are <em>tasty</em>.
</ins>
<ins datetime="2007-12-19 00:00Z">
  So are pears.
</ins>
</aside>
```

#### 4.7.2 L' ~~élément~~



MDN

##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Transparente](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
[cite](#)— Lien vers la source de la citation ou plus d'informations sur la modification  
[datetime](#)— Date et (facultativement) heure du changement

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

Utilisations [HTMLModElement](#).

L' ~~élément~~ [représente](#) une suppression du document.

~~les~~ éléments ne doivent pas dépasser les limites [implicites des paragraphes](#) .

Ce qui suit montre une liste "à faire" où les éléments qui ont été faits sont barrés avec la date et l'heure de leur achèvement.

```
<h1>To Do</h1>
<ul>
  <li>Empty the dishwasher</li>
  <li><del datetime="2009-10-11T01:25-07:00">Watch Walter Lewin's
lectures</del></li>
  <li><del datetime="2009-10-10T23:38-07:00">Download more
tracks</del></li>
  <li>Buy a printer</li>
</ul>
```

#### 4.7.3 Attributs communs aux éléments ins et del

L' **cite** attribut peut être utilisé pour spécifier l' [URL](#) d'un document qui explique le changement. Lorsque ce document est long, par exemple le procès-verbal d'une réunion, les auteurs sont encouragés à inclure un [fragment](#) pointant vers la partie spécifique de ce document qui traite du changement.

Si l' **cite** attribut est présent, il doit s'agir d'une [URL valide potentiellement entourée d'espaces](#) expliquant le changement. Pour obtenir le lien de citation correspondant, la valeur de l'attribut doit être [analysée](#) par rapport au [nœud document](#) de l'élément . Les agents utilisateurs peuvent permettre aux utilisateurs de suivre ces liens de citation, mais ils sont principalement destinés à un usage privé (par exemple, par des scripts côté serveur collectant des statistiques sur les modifications d'un site), pas aux lecteurs.

L' **datetime** attribut peut être utilisé pour spécifier l'heure et la date du changement.

S'il est présent, la [datetime](#) valeur de l'attribut doit être une [chaîne de date valide avec time facultatif](#) .

Les agents utilisateurs doivent analyser l' **datetime** attribut selon l' algorithme [d'analyse d'une chaîne de date ou d'heure](#) . Si cela ne renvoie pas une [date](#) ou une [date et une heure globales](#) , alors la modification n'a pas d'horodatage associé (la valeur n'est pas conforme ; ce n'est pas une [chaîne de date valide avec une heure optionnelle](#) ). Dans le cas contraire, la modification est marquée comme ayant été effectuée à la [date](#) donnée ou [à la date et heure globales](#) . Si la valeur donnée est une [date et une heure globales](#) , les agents utilisateurs doivent utiliser les informations de décalage de fuseau horaire associées pour déterminer dans quel fuseau horaire présenter la date/heure donnée.

Cette valeur peut être montrée à l'utilisateur, mais elle est principalement destinée à un usage privé.

Les éléments ins et del doivent implémenter l' HTMLModElement interface :



```
[Exposed=Window]
```

```
interface HTMLModElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute USVString cite;
```

```
  [CEReactions] attribute DOMString dateTime;
```

```
};
```

L' cite attribut IDL doit refléter l'attribut de contenu de l'élément cite. L' dateTime attribut IDL doit refléter l'attribut de contenu de l'élément datetime.

## 4.7.4 Modifications et paragraphes

*Cette section est non normative.*

Étant donné que les éléments ins et del n'affectent pas la mise en paragraphe , il est possible, dans certains cas où les paragraphes sont implicites (sans p éléments explicites), qu'un élément ins ou del couvre à la fois un paragraphe entier ou d'autres éléments de contenu sans expression et une partie d'un autre paragraphe. Par exemple:

```
<section>
  <ins>
    <p>
      This is a paragraph that was inserted.
    </p>
    This is another paragraph whose first sentence was inserted
    at the same time as the paragraph above.
  </ins>
  This is a second sentence, which was there all along.
</section>
```

En n'enveloppant que certains paragraphes dans p des éléments, on peut même obtenir que la fin d'un paragraphe, un deuxième paragraphe entier et le début d'un troisième paragraphe soient couverts par le même élément ins ou del (bien que cela soit très déroutant et non considéré comme une bonne pratique ):

```
<section>
  This is the first paragraph. <ins>This sentence was
  inserted.
  <p>This second paragraph was inserted.</p>
  This sentence was inserted too.</ins> This is the
  third paragraph in this example.
  <!-- (don't do this) -->
</section>
```

Cependant, en raison de la façon dont [les paragraphes implicites](#) sont définis, il n'est pas possible de marquer la fin d'un paragraphe et le début du suivant en utilisant le même élément ins ou del. Il faut plutôt utiliser un (ou deux) p élément(s) et deux éléments ins or del, comme par exemple :del

```
<section>
  <p>This is the first paragraph. <del>This sentence was
  deleted.</del></p>
  <p><del>This sentence was deleted too.</del> That
  sentence needed a separate &lt;del> element.</p>
</section>
```

En partie à cause de la confusion décrite ci-dessus, les auteurs sont fortement encouragés à toujours baliser tous les paragraphes avec l' p élément, au lieu d'avoir ins ou del des éléments qui traversent les limites [implicites des paragraphes](#) .

## 4.7.5 Modifications et listes

*Cette section est non normative.*

Les modèles de contenu des éléments ol et ul n'autorisent pas les éléments ins et del en tant qu'enfants. Les listes représentent toujours tous leurs éléments, y compris les éléments qui auraient autrement été marqués comme supprimés.

Pour indiquer qu'un élément est inséré ou supprimé, un élément ins ou del peut entourer le contenu de l' li élément. Pour indiquer qu'un élément a été remplacé par

un autre, un même liélément peut avoir un ou plusieurs deléléments suivis d'un ou plusieurs inséléments.

Dans l'exemple suivant, une liste qui a commencé vide a eu des éléments ajoutés et supprimés au fil du temps. Les bits de l'exemple qui ont été soulignés montrent les parties qui sont l'état "actuel" de la liste. Cependant, les numéros d'éléments de la liste ne tiennent pas compte des modifications.

```
<h1>Stop-ship bugs</h1>
<ol>
  <li><ins datetime="2008-02-12T15:20Z">Bug 225:
    Rain detector doesn't work in snow</ins></li>
  <li><del datetime="2008-03-01T20:22Z"><ins datetime="2008-02-14T12:02Z">Bug 228:
    Water buffer overflows in April</ins></del></li>
  <li><ins datetime="2008-02-16T13:50Z">Bug 230:
    Water heater doesn't use renewable fuels</ins></li>
  <li><del datetime="2008-02-20T21:15Z"><ins datetime="2008-02-16T14:25Z">Bug 232:
    Carbon dioxide emissions detected after startup</ins></del></li>
</ol>
```

Dans l'exemple suivant, une liste commençant uniquement par des fruits a été remplacée par une liste contenant uniquement des couleurs.

```
<h1>List of <del>fruits</del><ins>colors</ins></h1>
<ul>
  <li><del>Lime</del><ins>Green</ins></li>
  <li><del>Apple</del></li>
  <li>Orange</li>
  <li><del>Pear</del></li>
  <li><ins>Teal</ins></li>
  <li><del>Lemon</del><ins>Yellow</ins></li>
  <li>Olive</li>
  <li><ins>Purple</ins></li>
</ul>
```

## 4.7.6 Modifications et tableaux

*Cette section est non normative.*

Les éléments qui font partie du modèle de table ont des exigences de modèle de contenu compliquées qui n'autorisent pas les éléments `ins` et `del`, il peut donc être difficile d'indiquer des modifications à une table.

Pour indiquer qu'une ligne entière ou une colonne entière a été ajoutée ou supprimée, le contenu entier de chaque cellule de cette ligne ou colonne peut être enveloppé dans des éléments `ins` ou `del` (respectivement).

Ici, une ligne de tableau a été ajoutée :

```
<table>
  <thead>
    <tr> <th> Game name          <th> Game publisher    <th> Verdict
  <tbody>
    <tr> <td> Diablo 2            <td> Blizzard      <td> 8/10
    <tr> <td> Portal              <td> Valve          <td> 10/10
    <tr> <td> <ins>Portal 2</ins> <td> <ins>Valve</ins> <td>
<ins>10/10</ins>
  </table>
```

Ici, une colonne a été supprimée (l'heure à laquelle elle a été supprimée est également indiquée, ainsi qu'un lien vers la page expliquant pourquoi) :

```
<table>
  <thead>
    <tr> <th> Game name          <th> Game publisher    <th> <del
cite="/edits/r192" datetime="2011-05-02 14:23Z">Verdict</del>
  <tbody>
    <tr> <td> Diablo 2            <td> Blizzard      <td> <del
cite="/edits/r192" datetime="2011-05-02 14:23Z">8/10</del>
    <tr> <td> Portal              <td> Valve          <td> <del
cite="/edits/r192" datetime="2011-05-02 14:23Z">10/10</del>
    <tr> <td> Portal 2            <td> Valve          <td> <del
cite="/edits/r192" datetime="2011-05-02 14:23Z">10/10</del>
  </table>
```

De manière générale, il n'y a pas de bon moyen d'indiquer des modifications plus compliquées (par exemple, qu'une cellule a été supprimée, en déplaçant toutes les cellules suivantes vers le haut ou vers la gauche).

## 4.8 Contenu intégré

### 4.8.1 L' `picture` élément





## Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu intégré](#) .  
[Contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

## Modèle de contenu :

Zéro ou plusieurs [source](#) éléments, suivis d'un [img](#) élément, éventuellement mélangés à [des éléments de support de script](#) .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

[Attributs globaux](#)

## Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

## Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLPictureElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
};
```

L' [picture](#) élément est un conteneur qui fournit plusieurs sources à son [img](#) élément contenu pour permettre aux auteurs de contrôler de manière déclarative ou de donner des indications à l'agent utilisateur sur la ressource d'image à utiliser, en fonction de la densité de pixels de l'écran, de la taille [de la fenêtre d'affichage](#) , du format de l'image et d'autres facteurs. Il [représente](#) ses enfants.

*L' [picture](#) élément est quelque peu différent des éléments similaires [video](#) et [audio](#). Bien qu'ils contiennent tous des [source](#) éléments, l'attribut [source](#) de l'élément [src](#) n'a aucune signification lorsque l'élément est imbriqué dans un [picture](#) élément et que l'algorithme de sélection des ressources est différent. De plus, l' [picture](#) élément lui-même n'affiche rien ; il fournit simplement un contexte pour son [img](#) élément contenu qui lui permet de choisir parmi plusieurs [URL](#) .*

## 4.8.2 L' **source** élément



### Catégories :

Aucun.

### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un [picture](#) élément, avant l' [img](#) élément.

En tant qu'enfant d'un [élément média](#) , avant tout [contenu](#) ou [track](#) élément de flux.

### Modèle de contenu :

[Rien](#) .

### Omission de balise dans text/html :

Pas [de balise de fin](#) .

### Attributs de contenu :

#### Attributs globaux

[type](#) — Type de ressource intégrée

[src](#)(en [video](#) ou [audio](#)) — Adresse de la ressource

[srcset](#)(dans [picture](#)) — Images à utiliser dans différentes situations, par exemple, écrans haute résolution, petits moniteurs, etc.

[sizes](#)(en [picture](#)) — Tailles d'image pour différentes mises en page

[media](#)(en [picture](#)) — Supports applicables

[width](#)(po [picture](#)) — dimension horizontale

[height](#)(po [picture](#)) — Dimension verticale

### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### Interface DOM :

[Exposed=Window]

```
interface HTMLSourceElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute USVString src;
```

```
    [CEReactions] attribute DOMString type;
```

```
    [CEReactions] attribute USVString srcset;
```

```

[CEReactions] attribute DOMString sizes;

[CEReactions] attribute DOMString media;

[CEReactions] attribute unsigned long width;

[CEReactions] attribute unsigned long height;

};

```

L' [source](#) élément permet aux auteurs de spécifier plusieurs [ensembles de sources](#) alternatives pour [img](#) les éléments ou plusieurs [ressources multimédias](#) alternatives pour [les éléments multimédias](#) . Il ne [représente](#) rien en soi.

L' [type](#) attribut peut être présent. Si elle est présente, la valeur doit être une [chaîne de type MIME valide](#) .

Le reste des exigences dépend de si le parent est un [picture](#) élément ou un [élément média](#) :

#### Le [source](#) parent de l'élément est un [picture](#) élément

L' [srcset](#) attribut doit être présent et est un [attribut srcset](#) .

L' [srcset](#) attribut ajoute les [sources d'image](#) à l' [ensemble source](#) , si l' [source](#) élément est sélectionné.

Si l' [srcset](#) attribut a des [chaînes candidates d'image](#) utilisant un [descripteur de largeur](#) , l' [sizes](#) attribut doit également être présent et est un [attribut de tailles](#) . L' [sizes](#) attribut contribue à la [taille source](#) de l' [ensemble source](#) , si l' [source](#) élément est sélectionné.

L' [media](#) attribut peut également être présent. Si elle est présente, la valeur doit contenir une [liste de requêtes multimédia valide](#) . L'agent utilisateur passera à l' [source](#) élément suivant si la valeur ne [correspond pas à l'environnement](#) .

L' [source](#) élément prend en charge [les attributs de dimension](#) . L' [img](#) élément peut utiliser les attributs [width](#) et [height](#) d'un [source](#) élément, au lieu de ceux de l' [img](#) élément lui-même, pour déterminer ses dimensions de rendu et son rapport d'aspect, [comme défini dans la section Rendu](#) .

L' [type](#) attribut donne le type des images dans l' [ensemble source](#) , pour permettre à l'agent utilisateur de passer à l' [source](#) élément suivant s'il ne prend pas en charge le type donné.

Si l' type attribut n'est pas spécifié, l'agent utilisateur ne sélectionnera pas un source élément différent s'il constate qu'il ne prend pas en charge le format d'image après l'avoir récupéré.

Lorsqu'un source élément a un source élément frère ou img un élément suivant avec un srcset attribut spécifié, il doit avoir au moins l'un des éléments suivants :

- Un media attribut spécifié avec une valeur qui, après suppression des espaces blancs ASCII de début et de fin , n'est pas la chaîne vide et n'est pas une correspondance ASCII insensible à la casse pour la chaîne " all".
- Un type attribut spécifié.

L' src attribut ne doit pas être présent.

### Le source parent de l'élément est un élément multimédia

L' src attribut donne l' URL de la ressource média . La valeur doit être une URL non vide valide potentiellement entourée d'espaces . Cet attribut doit être présent.

L' type attribut donne le type de la ressource média , pour aider l'agent utilisateur à déterminer s'il peut lire cette ressource média avant de la récupérer. Le codecs paramètre, défini par certains types MIME, peut être nécessaire pour spécifier exactement comment la ressource est encodée. [\[RFC6381\]](#)

*La modification dynamique de l'attribut src d'un source élément lorsque l'élément est déjà inséré dans un élément or n'aura aucun effet. Pour changer ce qui se joue, il suffit d'utiliser directement l' attribut sur l' élément média , en utilisant éventuellement la méthode pour choisir parmi les ressources disponibles. Généralement, la manipulation manuelle des éléments après l'analyse du document est une approche inutilement compliquée.*

*src type video audio src canPlayType () source*

La liste suivante montre quelques exemples d'utilisation du codecs=paramètre MIME dans l' type attribut.

**Vidéo de profil de base H.264 contraint (compatible vidéo principale et étendue) niveau 3 et audio AAC à faible complexité dans un conteneur MP4**

```
<source src='video.mp4' type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>
```

**Vidéo à profil étendu H.264 (compatible avec la ligne de base) niveau 3 et audio AAC à faible complexité dans un conteneur MP4**

```
<source src='video.mp4' type='video/mp4; codecs="avc1.58A01E, mp4a.40.2"'>
```

**Vidéo de profil principal H.264 de niveau 3 et audio AAC à faible complexité dans un conteneur MP4**

```
<source src='video.mp4' type='video/mp4; codecs="avc1.4D401E, mp4a.40.2"'>
```

**Vidéo de profil H.264 "élevé" (incompatible avec les profils principaux, de base ou étendus) niveau 3 et audio AAC à faible complexité dans le conteneur MP4**

```
<source src='video.mp4' type='video/mp4; codecs="avc1.64001E, mp4a.40.2"'>
```

**Vidéo MPEG-4 Visual Simple Profile Level 0 et audio AAC à faible complexité dans un conteneur MP4**

```
<source src='video.mp4' type='video/mp4; codecs="mp4v.20.8, mp4a.40.2"'>
```

**Vidéo MPEG-4 Advanced Simple Profile Level 0 et audio AAC à faible complexité dans un conteneur MP4**

```
<source src='video.mp4' type='video/mp4; codecs="mp4v.20.240, mp4a.40.2"'>
```

**Vidéo MPEG-4 Visual Simple Profile Level 0 et audio AMR dans un conteneur 3GPP**

```
<source src='video.3gp' type='video/3gpp; codecs="mp4v.20.8, samr"'>
```

**Vidéo Theora et audio Vorbis dans un conteneur Ogg**

```
<source src='video.ogv' type='video/ogg; codecs="theora, vorbis"'>
```

**Vidéo Theora et audio Speex dans un conteneur Ogg**

```
<source src='video.ogv' type='video/ogg; codecs="theora, speex"'>
```

**Audio Vorbis seul dans un conteneur Ogg**

```
<source src='audio.ogg' type='audio/ogg; codecs=vorbis'>
```

**Audio Speex seul dans un conteneur Ogg**

```
<source src='audio.spx' type='audio/ogg; codecs=speex'>
```

**Audio FLAC seul dans un conteneur Ogg**

```
<source src='audio.oga' type='audio/ogg; codecs=flac'>
```

**Vidéo Dirac et audio Vorbis dans un conteneur Ogg**

```
<source src='video.ogv' type='video/ogg; codecs="dirac, vorbis"'>
```

Les attributs [srcset](#), [sizes](#) et [media](#) ne doivent pas être présents.

Les [source](#) [étapes d'insertion de l'élément HTML](#) , étant donné *insertNode* , sont :

1. Si le parent de *insertNode* est un [élément média](#) qui n'a pas [src](#) d'attribut et dont [networkState](#) a la valeur [NETWORK\\_EMPTY](#) , alors invoquez l' [algorithme de sélection de ressource](#) de cet [élément média](#) .

2. Si le frère suivant de *insertNode* est un imgélément et que son parent est un pictureélément, alors, comptez cela comme une mutation pertinente pour l' imgélément.

Les source étapes de suppression de l'élément HTML , étant donnés *removeNode* et *oldParent* , sont :

1. Si le frère suivant de *removeNode* était un imgélément et *oldParent* est un pictureélément, alors, comptez cela comme une mutation pertinente pour l' imgélément.

Les attributs IDL **src**, **type**, et doivent refléter**srcset** les attributs de contenu respectifs du même nom.**sizesmedia**

Si l'auteur n'est pas sûr que les agents utilisateurs seront tous en mesure de restituer les ressources multimédias fournies, l'auteur peut écouter l' errorévénement sur le dernier sourceélément et déclencher un comportement de secours :

```
<script>
function fallback(video) {
    // replace <video> with its contents
    while (video.hasChildNodes()) {
        if (video.firstChild instanceof HTMLSourceElement)
            video.removeChild(video.firstChild);
        else
            video.parentNode.insertBefore(video.firstChild, video);
    }
    video.parentNode.removeChild(video);
}
</script>
<video controls autoplay>
  <source src='video.mp4' type='video/mp4; codecs="avc1.42E01E, mp4a.40.2"'>
  <source src='video.ogv' type='video/ogg; codecs="theora, vorbis"'
    onerror="fallback(parentNode)">
  ...
</video>
```

#### 4.8.3 L' **img**élément



## Catégories :

[Contenu du flux](#) .

[Contenu de la phrase](#) .

[Contenu intégré](#) .

[Élément associé au formulaire](#) .

Si l'élément a un [usemap](#) attribut : [Contenu interactif](#) .

[Contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

## Modèle de contenu :

[Rien](#) .

## Omission de balise dans text/html :

Pas [de balise de fin](#) .

## Attributs de contenu :

[Attributs globaux](#)

[alt](#)— Texte de remplacement à utiliser lorsque les images ne sont pas disponibles

[src](#)— Adresse de la ressource

[srcset](#)— Images à utiliser dans différentes situations, par exemple, écrans haute résolution, petits moniteurs, etc.

[sizes](#)— Tailles d'image pour différentes mises en page

[crossorigin](#)— Comment l'élément gère les requêtes crossorigin

[usemap](#)— Nom de [l'image cliquable](#) à utiliser

[ismap](#)— Si l'image est une image cliquable côté serveur

[width](#)— Dimensions horizontales

[height](#)— Dimension verticale

[referrerpolicy](#)— [Politique de référence](#) pour [les récupérations](#) initiées par l'élément

[decoding](#)— Indice de décodage à utiliser lors du traitement de cette image pour la présentation

[loading](#)— Utilisé lors de la détermination du report de chargement

## Considérations d'accessibilité :

[alt](#) Si l'élément a un attribut non vide : [pour les auteurs](#) ; [pour les exécutants](#) .

Sinon : [pour les auteurs](#) ; [pour les exécutants](#) .

## Interface DOM :

```
[Exposed=Window,
```

```
  LegacyFactoryFunction=Image(optional unsigned long width,
```

```
  optional unsigned long height)]
```

```
interface HTMLImageElement : HTMLElement {
```

```
[HTMLConstructor] constructor();
```

```
[CEReactions] attribute DOMString alt;
```

```
[CEReactions] attribute USVString src;
```

```
[CEReactions] attribute USVString srcset;
```

```
[CEReactions] attribute DOMString sizes;
```

```
[CEReactions] attribute DOMString? crossOrigin;
```

```
[CEReactions] attribute DOMString useMap;
```

```
[CEReactions] attribute boolean isMap;
```

```
[CEReactions] attribute unsigned long width;
```

```
[CEReactions] attribute unsigned long height;
```

```
readonly attribute unsigned long naturalWidth;
```

```
readonly attribute unsigned long naturalHeight;
```

```
readonly attribute boolean complete;
```

```
readonly attribute USVString currentSrc;
```

```
[CEReactions] attribute DOMString referrerPolicy;
```

```
[CEReactions] attribute DOMString decoding;
```

```
[CEReactions] attribute DOMString loading;
```

```
Promise<undefined> decode();
```

```
// also has obsolete members
```

```
};
```

Un [img](#) élément représente une image.



Un `img` élément a un **attribut de dimension source** , initialement défini sur l'élément lui-même.



L'image donnée par les attributs `src` et `srcset`, et tous les attributs `source` des éléments frères précédents `srcset` si le parent est un `picture` élément, est le contenu incorporé ; la valeur de l' `alt` attribut fournit un contenu équivalent pour ceux qui ne peuvent pas traiter les images ou dont le chargement des images est désactivé (c'est-à-dire qu'il s'agit du [contenu de secours](#) `img` de l'élément ).

Les exigences relatives à la `alt` valeur de l'attribut sont décrites [dans une section distincte](#) .

L' `src` attribut doit être présent et doit contenir une [URL non vide valide potentiellement entourée d'espaces](#) faisant référence à une ressource d'image non interactive, éventuellement animée, qui n'est ni paginée ni scriptée.

*Les exigences ci-dessus impliquent que les images peuvent être des bitmaps statiques (par exemple, PNG, GIF, JPEG), des documents vectoriels d'une seule page (PDF d'une seule page, fichiers XML avec un élément de document SVG), des bitmaps animés (APNG, GIF animés), des graphiques (fichiers XML avec un [élément de document](#) SVG qui utilise une animation SMIL déclarative), etc. Cependant, ces définitions excluent les fichiers SVG avec script, les fichiers PDF multipages, les fichiers MNG interactifs, les documents HTML, les documents en texte brut, etc. [\[PNG\]](#) [\[GIF\]](#) [\[JPEG\]](#) [\[PDF\]](#) [\[XML\]](#) [\[APNG\]](#) [\[SVG\]](#) [\[MNG\]](#)*

L' `srcset` attribut peut également être présent et est un [attribut srcset](#) .

L' `srcset` attribut et l' `src` attribut (si [les descripteurs de largeur](#) ne sont pas utilisés) contribuent les [sources d'image](#) à l' [ensemble source](#) (si aucun `source` élément n'a été sélectionné).

Si l' `srcset` attribut est présent et a des [chaînes candidates d'image](#) utilisant un [descripteur de largeur](#) , l' `sizes` attribut doit également être présent et est un [attribut de tailles](#) . L' `sizes` attribut contribue la [taille de la source](#) à l' [ensemble source](#) (si aucun `source` élément n'a été sélectionné).



The `crossorigin` attribute is a [CORS settings attribute](#). Its purpose is to allow images from third-party sites that allow cross-origin access to be used with [canvas](#).

The `referrerpolicy` attribute is a [referrer policy attribute](#). Its purpose is to set the [referrer policy](#) used when [fetching](#) the image. [\[REFERRERPOLICY\]](#)

The **decoding** attribute indicates the preferred method to [decode](#) this image. The attribute, if present, must be an [image decoding hint](#). This attribute's [missing value default](#) and [invalid value default](#) are both the [auto](#) state.

The **loading** attribute is a [lazy loading attribute](#). Its purpose is to indicate the policy for loading images that are outside the viewport.

When the **loading** attribute's state is changed to the [Eager](#) state, the user agent must run these steps:

1. Let *resumptionSteps* be the [img](#) element's [lazy load resumption steps](#).
2. If *resumptionSteps* is null, then return.
3. Set the [img](#)'s [lazy load resumption steps](#) to null.
4. Invoke *resumptionSteps*.

```



<div id=very-large></div> <!-- Everything after this div is below
the viewport -->


```

In the example above, the images load as follows:

1. jpeg, 2. jpeg, 4. jpeg

The images load eagerly and delay the window's load event.

3. jpeg

The image loads when layout is known, due to being in the viewport, however it does not delay the window's load event.

5. jpeg

The image loads only once scrolled into the viewport, and does not delay the window's load event.

*Developers are encouraged to specify an intrinsic aspect ratio via [width](#) and [height](#) attributes on lazy loaded images, even if CSS sets the image's width and height properties, to prevent the page layout from shifting around after the image loads.*

The [img](#) [HTML element insertion steps](#), given *insertedNode*, are:

1. If *insertedNode*'s parent is a [picture](#) element, then, count this as a [relevant mutation](#) for *insertedNode*.

The [img HTML element removing steps](#), given *removedNode* and *oldParent*, are:

1. If *oldParent* is a [picture](#) element, then, count this as a [relevant mutation](#) for *removedNode*.
- 

The [img](#) element must not be used as a layout tool. In particular, [img](#) elements should not be used to display transparent images, as such images rarely convey meaning and rarely add anything useful to the document.

---

What an [img](#) element represents depends on the [src](#) attribute and the [alt](#) attribute.

**If the [src](#) attribute is set and the [alt](#) attribute is set to the empty string**

The image is either decorative or supplemental to the rest of the content, redundant with some other information in the document.

If the image is [available](#) and the user agent is configured to display that image, then the element [represents](#) the element's image data.

Otherwise, the element [represents](#) nothing, and may be omitted completely from the rendering. User agents may provide the user with a notification that an image is present but has been omitted from the rendering.

**If the [src](#) attribute is set and the [alt](#) attribute is set to a value that isn't empty**

The image is a key part of the content; the [alt](#) attribute gives a textual equivalent or replacement for the image.

If the image is [available](#) and the user agent is configured to display that image, then the element [represents](#) the element's image data.

Otherwise, the element [represents](#) the text given by the [alt](#) attribute. User agents may provide the user with a notification that an image is present but has been omitted from the rendering.

**If the [src](#) attribute is set and the [alt](#) attribute is not**

The image might be a key part of the content, and there is no textual equivalent of the image available.

*In a conforming document, the absence of the alt attribute indicates that the image is a key part of the content but that a textual replacement for the image was not available when the image was generated.*

If the image is available and the user agent is configured to display that image, then the element represents the element's image data.

If the image has a src attribute whose value is the empty string, then the element represents nothing.

Otherwise, the user agent should display some sort of indicator that there is an image that is not being rendered, and may, if requested by the user, or if so configured, or when required to provide contextual information in response to navigation, provide caption information for the image, derived as follows:

1. If the image has a title attribute whose value is not the empty string, then return the value of that attribute.
2. If the image is a descendant of a figure element that has a child figcaption element, and, ignoring the figcaption element and its descendants, the figure element has no flow content descendants other than inter-element whitespace and the img element, then return the contents of the first such figcaption element.
3. Return nothing. (There is no caption information.)

**If the src attribute is not set and either the alt attribute is set to the empty string or the alt attribute is not set at all**

The element represents nothing.

**Otherwise**

The element represents the text given by the alt attribute.

The alt attribute does not represent advisory information. User agents must not present the contents of the alt attribute in the same way as content of the title attribute.

User agents may always provide the user with the option to display any image, or to prevent any image from being displayed. User agents may also apply heuristics to help the user make use of the image when the user is unable to see it, e.g. due to a visual disability or because they are using a text terminal with no graphics capabilities. Such heuristics could include, for instance, optical character recognition (OCR) of text found within the image.

***While user agents are encouraged to repair cases of missing alt attributes, authors must not rely on such behavior. Requirements for providing text to act as an alternative for images are described in detail below.***

The *contents* of [img](#) elements, if any, are ignored for the purposes of rendering.

---

The [usemap](#) attribute, if present, can indicate that the image has an associated [image map](#).

The [ismap](#) attribute, when used on an element that is a descendant of an [a](#) element with an [href](#) attribute, indicates by its presence that the element provides access to a server-side image map. This affects how events are handled on the corresponding [a](#) element.

The [ismap](#) attribute is a [boolean attribute](#). The attribute must not be specified on an element that does not have an ancestor [a](#) element with an [href](#) attribute.

*The [usemap](#) and [ismap](#) attributes can result in confusing behavior when used together with [source](#) elements with the [media](#) attribute specified in a [picture](#) element.*

The [img](#) element supports [dimension attributes](#).



The [alt](#), [src](#), [srcset](#) and [sizes](#) IDL attributes must [reflect](#) the respective content attributes of the same name.



The [crossOrigin](#) IDL attribute must [reflect](#) the [crossorigin](#) content attribute, [limited to only known values](#).



The [useMap](#) IDL attribute must [reflect](#) the [usemap](#) content attribute.



The [isMap](#) IDL attribute must [reflect](#) the [ismap](#) content attribute.



The [referrerPolicy](#) IDL attribute must [reflect](#) the [referrerpolicy](#) content attribute, [limited to only known values](#).

✓MDN

The **decoding** IDL attribute must [reflect](#) the **decoding** content attribute, [limited to only known values](#).

✓MDN

The **loading** IDL attribute must [reflect](#) the **loading** content attribute, [limited to only known values](#).

**image.width** [ = value ]

✓MDN

**image.height** [ = value ]

✓MDN

These attributes return the actual rendered dimensions of the image, or zero if the dimensions are not known.

They can be set, to change the corresponding content attributes.

**image.naturalWidth**

✓MDN

**image.naturalHeight**

✓MDN

These attributes return the intrinsic dimensions of the image, or zero if the dimensions are not known.

**image.complete**

✓MDN

Returns true if the image has been completely downloaded or if no image is specified; otherwise, returns false.

**image.currentSrc**

✓MDN

Returns the image's [absolute URL](#).

**image.decode** ()

✓MDN

This method causes the user agent to [decode](#) the image [in parallel](#), returning a promise that fulfills when decoding is complete.

The promise will be rejected with an ["EncodingError"](#) [DOMException](#) if the image cannot be decoded.

**image** = new **Image** ([ width [, height ] ])

✓MDN

Returns a new `img` element, with the `width` and `height` attributes set to the values passed in the relevant arguments, if applicable.

The IDL attributes `width` and `height` must return the rendered width and height of the image, in [CSS pixels](#), if the image is [being rendered](#), and is being rendered to a visual medium; or else the [density-corrected intrinsic width and height](#) of the image, in [CSS pixels](#), if the image has [intrinsic dimensions](#) and is [available](#) but not being rendered to a visual medium; or else 0, if the image is not [available](#) or does not have [intrinsic dimensions](#). [\[CSS\]](#)

On setting, they must act as if they [reflected](#) the respective content attributes of the same name.

The IDL attributes `naturalWidth` and `naturalHeight` must return the [density-corrected intrinsic width and height](#) of the image, in [CSS pixels](#), if the image has [intrinsic dimensions](#) and is [available](#), or else 0. [\[CSS\]](#)

*Since the [intrinsic dimensions](#) of an image take into account any orientation specified in its metadata, `naturalWidth` and `naturalHeight` reflect the dimensions after applying any rotation needed to correctly orient the image, regardless of the value of the ['image-orientation'](#) property.*

The IDL attribute `complete` must return true if any of the following conditions is true:

- Both the `src` attribute and the `srcset` attribute are omitted.
- The `srcset` attribute is omitted and the `src` attribute's value is the empty string.
- The `img` element's [current request](#)'s `state` is [completely available](#) and its [pending request](#) is null.
- The `img` element's [current request](#)'s `state` is [broken](#) and its [pending request](#) is null.

Otherwise, the attribute must return false.

The `currentSrc` IDL attribute must return the `img` element's [current request](#)'s [current URL](#).

The `decode()` method, when invoked, must perform the following steps:

1. Let *promise* be a new promise.
2. [Queue a microtask](#) to perform the following steps:

*This is done because [updating the image data](#) takes place in a microtask as well. Thus, to make code such as*

```
img.src = "stars.jpg";  
img.decode();
```

*properly decode stars.jpg, we need to delay any processing by one microtask.*

1. If any of the following conditions are true about this `img` element:

- its `node document` is not `fully active`;
- its `current request`'s `state` is `broken`,

then reject *promise* with an `"EncodingError" DOMException`.

2. Otherwise, `in parallel`, wait for one of the following cases to occur, and perform the corresponding actions:

This `img` element's `node document` stops being `fully active`

This `img` element's `current request` changes or is mutated

This `img` element's `current request`'s `state` becomes `broken`

Reject *promise* with an `"EncodingError" DOMException`.

This `img` element's `current request`'s `state` becomes `completely available`

`Decode` the image.

If decoding does not need to be performed for this image (for example because it is a vector graphic), resolve *promise* with undefined.

If decoding fails (for example due to invalid image data), reject *promise* with an `"EncodingError" DOMException`.

If the decoding process completes successfully, resolve *promise* with undefined.

User agents should ensure that the decoded media data stays readily available until at least the end of the next successful `update the rendering` step in the `event loop`. This is an important part of the API contract, and should not be broken if at all possible. (Typically, this would only be violated in low-memory situations that require evicting decoded image data, or when the image is too large to keep in decoded form for this period of time.)

*Animated images will become `completely available` only after all their frames are loaded. Thus, even though an implementation could decode the first frame before that point, the above steps will not do so, instead waiting until all frames are available.*

3. Return *promise*.

Without the `decode()` method, the process of loading an `img` element and then displaying it might look like the following:

```
const img = new Image();
```



```
img.src = "nebula.jpg";
img.onload = () => {
  document.body.appendChild(img);
};
img.onerror = () => {
  document.body.appendChild(new Text("Could not load the nebula
:("));
};
```

However, this can cause notable dropped frames, as the paint that occurs after inserting the image into the DOM causes a synchronous decode on the main thread.

This can instead be rewritten using the [decode\(\)](#) method:

```
const img = new Image();
img.src = "nebula.jpg";
img.decode().then(() => {
  document.body.appendChild(img);
}).catch(() => {
  document.body.appendChild(new Text("Could not load the nebula
:("));
});
```

This latter form avoids the dropped frames of the original, by allowing the user agent to decode the image [in parallel](#), and only inserting it into the DOM (and thus causing it to be painted) once the decoding process is complete.

Because the [decode\(\)](#) method attempts to ensure that the decoded image data is available for at least one frame, it can be combined with the [requestAnimationFrame\(\)](#) API. This means it can be used with coding styles or frameworks that ensure that all DOM modifications are batched together as [animation frame callbacks](#):

```
const container = document.querySelector("#container");

const { containerWidth, containerHeight } = computeDesiredSize();
requestAnimationFrame(() => {
  container.style.width = containerWidth;
  container.style.height = containerHeight;
});

// ...

const img = new Image();
img.src = "supernova.jpg";
```

```
img.decode().then(() => {
  requestAnimationFrame(() => container.appendChild(img));
});
```

A legacy factory function is provided for creating [HTMLImageElement](#) objects (in addition to the factory methods from DOM such as [createElement\(\)](#)): [Image\(width, height\)](#). When invoked, the legacy factory function must perform the following steps:

1. Let *document* be the [current global object](#)'s [associated Document](#).
2. Let *img* be the result of [creating an element](#) given *document*, [img](#), and the [HTML namespace](#).
3. If *width* is given, then [set an attribute value](#) for *img* using "[width](#)" and *width*.
4. If *height* is given, then [set an attribute value](#) for *img* using "[height](#)" and *height*.
5. Return *img*.

A single image can have different appropriate alternative text depending on the context.

In each of the following cases, the same image is used, yet the [alt](#) text is different each time. The image is the coat of arms of the Carouge municipality in the canton Geneva in Switzerland.

Here it is used as a supplementary icon:

```
<p>I lived in  Carouge.</p>
```

Here it is used as an icon representing the town:

```
<p>Home town: </p>
```

Here it is used as part of a text on the town:

```
<p>Carouge has a coat of arms.</p>
<p></p>
<p>It is used as decoration all over the town.</p>
```

Here it is used as a way to support a similar text where the description is given as well as, instead of as an alternative to, the image:

```
<p>Carouge has a coat of arms.</p>
<p></p>
<p>The coat of arms depicts a lion, sitting in front of a tree.
```

```
It is used as decoration all over the town.</p>
```

Here it is used as part of a story:

```
<p>She picked up the folder and a piece of paper fell out.</p>
<p></p>
<p>She stared at the folder. S! The answer she had been looking for
all
this time was simply the letter S! How had she not seen that
before? It all
came together now. The phone call where Hector had referred to a
lion's tail,
the time Maria had stuck her tongue out...</p>
```

Here it is not known at the time of publication what the image will be, only that it will be a coat of arms of some kind, and thus no replacement text can be provided, and instead only a brief caption for the image is provided, in the title attribute:

```
<p>The last user to have uploaded a coat of arms uploaded this
one:</p>
<p></p>
```

Ideally, the author would find a way to provide real replacement text even in this case, e.g. by asking the previous user. Not providing replacement text makes the document more difficult to use for people who are unable to view images, e.g. blind users, or users or very low-bandwidth connections or who pay by the byte, or users who are forced to use a text-only web browser.

Here are some more examples showing the same picture used in different contexts, with different appropriate alternate texts each time.

```
<article>
  <h1>My cats</h1>
  <h2>Fluffy</h2>
  <p>Fluffy is my favorite.</p>
  
  <p>She's just too cute.</p>
  <h2>Miles</h2>
  <p>My other cat, Miles just eats and sleeps.</p>
</article>
<article>
  <h1>Photography</h1>
```

```

<h2>Shooting moving targets indoors</h2>

<p>The trick here is to know how to anticipate; to know at what
speed and
what distance the subject will pass by.</p>



<h2>Nature by night</h2>

<p>To achieve this, you'll need either an extremely sensitive
film, or
immense flash lights.</p>
</article>
<article>
  <h1>About me</h1>
  <h2>My pets</h2>
  <p>I've got a cat named Fluffy and a dog named Miles.</p>
  
  <p>My dog Miles and I like go on long walks together.</p>
  <h2>music</h2>
  <p>After our walks, having emptied my mind, I like listening to
Bach.</p>
</article>
<article>
  <h1>Fluffy and the Yarn</h1>
  <p>Fluffy was a cat who liked to play with yarn. She also liked to
jump.</p>
  <aside></aside>
  <p>She would play in the morning, she would play in the
evening.</p>
</article>

```

## 4.8.4 Images

### 4.8.4.1 Introduction

*This section is non-normative.*

To embed an image in HTML, when there is only a single image resource, use the [img](#) element and its [src](#) attribute.

```

<h2>From today's featured article</h2>

```

```

<p><b><a href="/wiki/Marie_Lloyd">Marie Lloyd</a></b> (1870-1922)
was an English <a href="/wiki/Music_hall">music hall</a> singer,
...
```

However, there are a number of situations for which the author might wish to use multiple image resources that the user agent can choose from:

- Different users might have different environmental characteristics:
  - The users' physical screen size might be different from one another.

A mobile phone's screen might be 4 inches diagonally, while a laptop's screen might be 14 inches diagonally.

4"14"

*This is only relevant when an image's rendered size depends on the [viewport](#) size.*

- The users' screen pixel density might be different from one another.

A mobile phone's screen might have three times as many physical pixels per inch compared to another mobile phone's screen, regardless of their physical screen size.

1x3x

- The users' zoom level might be different from one another, or might change for a single user over time.

A user might zoom in to a particular image to be able to get a more detailed look.

The zoom level and the screen pixel density (the previous point) can both affect the number of physical screen pixels per [CSS pixel](#). This ratio is usually referred to as **device-pixel-ratio**.

- The users' screen orientation might be different from one another, or might change for a single user over time.

A tablet can be held upright or rotated 90 degrees, so that the screen is either "portrait" or "landscape".

PortraitLandscape

- The users' network speed, network latency and bandwidth cost might be different from one another, or might change for a single user over time.

A user might be on a fast, low-latency and constant-cost connection while at work, on a slow, low-latency and constant-cost connection while at home, and on a variable-speed, high-latency and variable-cost connection anywhere else.

- Authors might want to show the same image content but with different rendered size depending on, usually, the width of the [viewport](#). This is usually referred to as **viewport-based selection**.

A web page might have a banner at the top that always spans the entire [viewport](#) width. In this case, the rendered size of the image depends on the physical size of the screen (assuming a maximised browser window).

Another web page might have images in columns, with a single column for screens with a small physical size, two columns for screens with medium physical size, and three columns for screens with big physical size, with the images varying in rendered size in each case to fill up the [viewport](#). In this case, the rendered size of an image might be *bigger* in the one-column layout compared to the two-column layout, despite the screen being smaller.

Narrow, 1 columnMedium, 2 columnsWide, 3 columns

- Authors might want to show different image content depending on the rendered size of the image. This is usually referred to as **art direction**.

When a web page is viewed on a screen with a large physical size (assuming a maximised browser window), the author might wish to include some less relevant parts surrounding the critical part of the image. When the same web page is viewed on a screen with a small physical size, the author might wish to show only the critical part of the image.

- Authors might want to show the same image content but using different image formats, depending on which image formats the user agent supports. This is usually referred to as **image format-based selection**.

A web page might have some images in the JPEG, WebP and JPEG XR image formats, with the latter two having better compression abilities compared to JPEG. Since different user agents can support different image formats, with some formats offering better compression ratios, the author would like to serve the better formats to user agents that support them, while providing JPEG fallback for user agents that don't.

The above situations are not mutually exclusive. For example, it is reasonable to combine different resources for different [device-pixel-ratio](#) with different resources for [art direction](#).

While it is possible to solve these problems using scripting, doing so introduces some other problems:

- Some user agents aggressively download images specified in the HTML markup, before scripts have had a chance to run, so that web pages complete loading sooner. If a script changes which image to download, the user agent will potentially start two separate downloads, which can instead cause worse page loading performance.
- If the author avoids specifying any image in the HTML markup and instead instantiates a single download from script, that avoids the double download problem above but then no image will be downloaded at all for users with scripting disabled and the aggressive image downloading optimization will also be disabled.

With this in mind, this specification introduces a number of features to address the above problems in a declarative manner.

### **Device-pixel-ratio-based selection when the rendered size of the image is fixed**

The [`src`](#) and [`srcset`](#) attributes on the [`img`](#) element can be used, using the `x` descriptor, to provide multiple images that only vary in their size (the smaller image is a scaled-down version of the bigger image).

*The `x` descriptor is not appropriate when the rendered size of the image depends on the [viewport](#) width ([viewport-based selection](#)), but can be used together with [art direction](#).*

```
<h2>From today's featured article</h2>

<p><b><a href="/wiki/Marie_Lloyd">Marie Lloyd</a></b> (1870-
1922)
was an English <a href="/wiki/Music_hall">music hall</a>
singer, ...
```

The user agent can choose any of the given resources depending on the user's screen's pixel density, zoom level, and possibly other factors such as the user's network conditions.

For backwards compatibility with older user agents that don't yet understand the [`srcset`](#) attribute, one of the URLs is specified in the [`img`](#) element's [`src`](#) attribute. This will result in something useful (though perhaps lower-resolution than the user would like) being displayed even in older user agents. For new user agents, the [`src`](#) attribute participates in the resource selection, as if it was specified in [`srcset`](#) with a `1x` descriptor.

The image's rendered size is given in the [`width`](#) and [`height`](#) attributes, which allows the user agent to allocate space for the image before it is downloaded.

### **Viewport-based selection**

The [srcset](#) and [sizes](#) attributes can be used, using the `w` descriptor, to provide multiple images that only vary in their size (the smaller image is a scaled-down version of the bigger image).

In this example, a banner image takes up the entire [viewport](#) width (using appropriate CSS).

```
<h1></h1>
```

The user agent will calculate the effective pixel density of each image from the specified `w` descriptors and the specified rendered size in the [sizes](#) attribute. It can then choose any of the given resources depending on the user's screen's pixel density, zoom level, and possibly other factors such as the user's network conditions.

If the user's screen is 320 [CSS pixels](#) wide, this is equivalent to specifying `wolf-400.jpg 1.25x`, `wolf-800.jpg 2.5x`, `wolf-1600.jpg 5x`. On the other hand, if the user's screen is 1200 [CSS pixels](#) wide, this is equivalent to specifying `wolf-400.jpg 0.33x`, `wolf-800.jpg 0.67x`, `wolf-1600.jpg 1.33x`. By using the `w` descriptors and the [sizes](#) attribute, the user agent can choose the correct image source to download regardless of how large the user's device is.

For backwards compatibility, one of the URLs is specified in the `img` element's [src](#) attribute. In new user agents, the [src](#) attribute is ignored when the [srcset](#) attribute uses `w` descriptors.

In this example, the web page has three layouts depending on the width of the [viewport](#). The narrow layout has one column of images (the width of each image is about 100%), the middle layout has two columns of images (the width of each image is about 50%), and the widest layout has three columns of images, and some page margin (the width of each image is about 33%). It breaks between these layouts when the [viewport](#) is 30em wide and 50em wide, respectively.

```

```

The [sizes](#) attribute sets up the layout breakpoints at 30em and 50em, and declares the image sizes between these breakpoints to be 100vw, 50vw, or `calc(33vw - 100px)`. These sizes do not necessarily have to match up exactly with the actual image width as specified in the CSS.

The user agent will pick a width from the [sizes](#) attribute, using the first item with a [<media-condition>](#) (the part in parentheses) that evaluates to true, or using the last item (`calc(33vw - 100px)`) if they all evaluate to false.



For example, if the [viewport](#) width is 29em, then `(max-width: 30em)` evaluates to true and 100vw is used, so the image size, for the purpose of resource selection, is 29em. If the [viewport](#) width is instead 32em, then `(max-width: 30em)` evaluates to false, but `(max-width: 50em)` evaluates to true and 50vw is used, so the image size, for the purpose of resource selection, is 16em (half the [viewport](#) width). Notice that the slightly wider [viewport](#) results in a smaller image because of the different layout.

The user agent can then calculate the effective pixel density and choose an appropriate resource similarly to the previous example.

### [Art direction](#)-based selection

The [picture](#) element and the [source](#) element, together with the [media](#) attribute, can be used to provide multiple images that vary the image content (for instance the smaller image might be a cropped version of the bigger image).

```
<picture>
  <source media="(min-width: 45em)" srcset="large.jpg">
  <source media="(min-width: 32em)" srcset="med.jpg">
  
</picture>
```

The user agent will choose the first [source](#) element for which the media query in the [media](#) attribute matches, and then choose an appropriate URL from its [srcset](#) attribute.

The rendered size of the image varies depending on which resource is chosen. To specify dimensions that the user agent can use before having downloaded the image, CSS can be used.

```
img { width: 300px; height: 300px }
```

```
@media (min-width: 32em) { img { width: 500px; height:300px }
```

```
} }
```

```
@media (min-width: 45em) { img { width: 700px; height:400px }
```

```
} }
```

This example combines [art direction](#)- and [device-pixel-ratio](#)-based selection. A banner that takes half the [viewport](#) is provided in two versions, one for wide screens and one for narrow screens.

```
<h1>
<picture>
  <source media="(max-width: 500px)" srcset="banner-
phone.jpeg, banner-phone-HD.jpeg 2x">
```

```

</picture>
</h1>
```

### Image format-based selection

The `type` attribute on the `source` element can be used to provide multiple images in different formats.

```
<h2>From today's featured article</h2>
<picture>
  <source srcset="/uploads/100-marie-lloyd.webp"
  type="image/webp">
  <source srcset="/uploads/100-marie-lloyd.jxr"
  type="image/vnd.ms-photo">
  
</picture>
<p><b><a href="/wiki/Marie_Lloyd">Marie Lloyd</a></b> (1870-
1922)
was an English <a href="/wiki/Music_hall">music hall</a>
singer, ...
```

In this example, the user agent will choose the first source that has a `type` attribute with a supported MIME type. If the user agent supports WebP images, the first `source` element will be chosen. If not, but the user agent does support JPEG XR images, the second `source` element will be chosen. If neither of those formats are supported, the `img` element will be chosen.

#### **4.8.4.1.1 Adaptive images**

*This section is non-normative.*

CSS and media queries can be used to construct graphical page layouts that adapt dynamically to the user's environment, in particular to different [viewport](#) dimensions and pixel densities. For content, however, CSS does not help; instead, we have the `img` element's `srcset` attribute and the `picture` element. This section walks through a sample case showing how to use these features.

Consider a situation where on wide screens (wider than 600 [CSS pixels](#)) a 300×150 image named `a-rectangle.png` is to be used, but on smaller screens (600 [CSS pixels](#) and less), a smaller 100×100 image called `a-square.png` is to be used. The markup for this would look like this:

```
<figure>
  <picture>
```

```

<source srcset="a-square.png" media="(max-width: 600px)">

</picture>
<figcaption>Barney Frank, 2011</figcaption>
</figure>

```

*For details on what to put in the [alt](#) attribute, see the [Requirements for providing text to act as an alternative for images](#) section.*

The problem with this is that the user agent does not necessarily know what dimensions to use for the image when the image is loading. To avoid the layout having to be reflowed multiple times as the page is loading, CSS and CSS media queries can be used to provide the dimensions:

```

<style>
#a { width: 300px; height: 150px; }
@media (max-width: 600px) { #a { width: 100px; height: 100px; } }
</style>
<figure>
<picture>
<source srcset="a-square.png" media="(max-width: 600px)">

</picture>
<figcaption>Barney Frank, 2011</figcaption>
</figure>

```

Alternatively, the [width](#) and [height](#) attributes can be used to provide the width and height for legacy user agents, using CSS just for the user agents that support [picture](#):

```

<style media="(max-width: 600px)">
#a { width: 100px; height: 100px; }
</style>
<figure>
<picture>
<source srcset="a-square.png" media="(max-width: 600px)">

</picture>
<figcaption>Barney Frank, 2011</figcaption>
</figure>

```

---

The [img](#) element is used with the [src](#) attribute, which gives the URL of the image to use for legacy user agents that do not support the [picture](#) element. This leads to a question of which image to provide in the [src](#) attribute.

If the author wants the biggest image in legacy user agents, the markup could be as follows:

```
<picture>
  <source srcset="pear-mobile.jpeg" media="(max-width: 720px)">
  <source srcset="pear-tablet.jpeg" media="(max-width: 1280px)">
  
</picture>
```

However, if legacy mobile user agents are more important, one can list all three images in the [source](#) elements, overriding the [src](#) attribute entirely.

```
<picture>
  <source srcset="pear-mobile.jpeg" media="(max-width: 720px)">
  <source srcset="pear-tablet.jpeg" media="(max-width: 1280px)">
  <source srcset="pear-desktop.jpeg">
  
</picture>
```

Since at this point the [src](#) attribute is actually being ignored entirely by [picture](#)-supporting user agents, the [src](#) attribute can default to any image, including one that is neither the smallest nor biggest:

```
<picture>
  <source srcset="pear-mobile.jpeg" media="(max-width: 720px)">
  <source srcset="pear-tablet.jpeg" media="(max-width: 1280px)">
  <source srcset="pear-desktop.jpeg">
  
</picture>
```

---

Above the `max-width` media feature is used, giving the maximum ([viewport](#)) dimensions that an image is intended for. It is also possible to use `min-width` instead.

```
<picture>
  <source srcset="pear-desktop.jpeg" media="(min-width: 1281px)">
  <source srcset="pear-tablet.jpeg" media="(min-width: 721px)">
  
</picture>
```

#### 4.8.4.2 Attributes common to [source](#), [img](#), and [link](#) elements

##### 4.8.4.2.1 Srcset attributes

A **srcset attribute** is an attribute with requirements defined in this section.

If present, its value must consist of one or more [image candidate strings](#), each separated from the next by a U+002C COMMA character (,). If an [image candidate string](#) contains no descriptors and no [ASCII whitespace](#) after the URL, the following [image candidate string](#), if there is one, must begin with one or more [ASCII whitespace](#).

An **image candidate string** consists of the following components, in order, with the further restrictions described below this list:

1. Zero or more [ASCII whitespace](#).
2. A [valid non-empty URL](#) that does not start or end with a U+002C COMMA character (,), referencing a non-interactive, optionally animated, image resource that is neither paged nor scripted.
3. Zero or more [ASCII whitespace](#).
4. Zero or one of the following:
  - A **width descriptor**, consisting of: [ASCII whitespace](#), a [valid non-negative integer](#) giving a number greater than zero representing the **width descriptor value**, and a U+0077 LATIN SMALL LETTER W character.
  - A **pixel density descriptor**, consisting of: [ASCII whitespace](#), a [valid floating-point number](#) giving a number greater than zero representing the **pixel density descriptor value**, and a U+0078 LATIN SMALL LETTER X character.
5. Zero or more [ASCII whitespace](#).

There must not be an [image candidate string](#) for an element that has the same [width descriptor value](#) as another [image candidate string](#)'s [width descriptor value](#) for the same element.

There must not be an [image candidate string](#) for an element that has the same [pixel density descriptor value](#) as another [image candidate string](#)'s [pixel density descriptor value](#) for the same element. For the purpose of this requirement, an [image candidate string](#) with no descriptors is equivalent to an [image candidate string](#) with a 1x descriptor.

If an [image candidate string](#) for an element has the [width descriptor](#) specified, all other [image candidate strings](#) for that element must also have the [width descriptor](#) specified.

The specified width in an [image candidate string](#)'s [width descriptor](#) must match the [intrinsic width](#) in the resource given by the [image candidate string](#)'s URL, if it has an [intrinsic width](#).

If an element has a [sizes attribute](#) present, all [image candidate strings](#) for that element must have the [width descriptor](#) specified.

#### 4.8.4.2.2 Sizes attributes

A **sizes attribute** is an attribute with requirements defined in this section.

If present, the value must be a [valid source size list](#).

A **valid source size list** is a string that matches the following grammar: [\[CSSVALUES\]](#) [\[MQ\]](#)

```
<source-size-list> = [ <source-size># , ]? <source-size-value>  
<source-size> = <media-condition> <source-size-value>  
<source-size-value> = <length>
```

A [<source-size-value>](#) must not be negative, and must not use CSS functions other than the [math functions](#).

The [<source-size-value>](#) gives the intended layout width of the image. The author can specify different widths for different environments with [<media-condition>](#)s.

*Percentages are not allowed in a [<source-size-value>](#), to avoid confusion about what it would be relative to. The '[vw](#)' unit can be used for sizes relative to the [viewport](#) width.*

#### 4.8.4.3 Processing model

---

An [img](#) element has a **current request** and a **pending request**. The [current request](#) is initially set to a new [image request](#). The [pending request](#) is initially set to null.

An **image request** has a **state**, **current URL**, and **image data**.

An [image request](#)'s [state](#) is one of the following:

***Unavailable***

The user agent hasn't obtained any image data, or has obtained some or all of the image data but hasn't yet decoded enough of the image to get the image dimensions.

***Partially available***

The user agent has obtained some of the image data and at least the image dimensions are available.

***Completely available***

The user agent has obtained all of the image data and at least the image dimensions are available.

***Broken***

The user agent has obtained all of the image data that it can, but it cannot even decode the image enough to get the image dimensions (e.g. the image is corrupted, or the format is not supported, or no data could be obtained).

An [image request](#)'s [current URL](#) is initially the empty string.

An [image request](#)'s [image data](#) is the decoded image data.

When an [image request](#)'s [state](#) is either [partially available](#) or [completely available](#), the [image request](#) is said to be **available**.

When an [img](#) element's [current request](#)'s [state](#) is [completely available](#) and the user agent can decode the media data without errors, then the [img](#) element is said to be **fully decodable**.

An [image request](#)'s [state](#) is initially [unavailable](#).

When an [img](#) element's [current request](#) is [available](#), the [img](#) element provides a [paint source](#) whose width is the image's [density-corrected intrinsic width](#) (if any), whose height is the image's [density-corrected intrinsic height](#) (if any), and whose appearance is the intrinsic appearance of the image.

---

An [img](#) element is said to **use srcset or picture** if it has a [srcset](#) attribute specified or if it has a parent that is a [picture](#) element.

---

Each [img](#) element has a **last selected source**, which must initially be null.

Each [image request](#) has a **current pixel density**, which must initially be 1.

Each [image request](#) has **preferred density-corrected dimensions**, which is either a struct consisting of a width and a height or is null. It must initially be null.

To determine the **density-corrected intrinsic width and height** of an [img](#) element *img*:

1. Let *dim* be *img*'s [current request](#)'s [preferred density-corrected dimensions](#).
2. If *dim* is null, set *dim* to *img*'s [intrinsic dimensions](#).
3. Set *dim*'s width to *dim*'s width divided by *img*'s [current request](#)'s [current pixel density](#).
4. Set *dim*'s height to *dim*'s height divided by *img*'s [current request](#)'s [current pixel density](#).
5. Return *dim*.

For example, if the [current pixel density](#) is 3.125, that means that there are 300 device pixels per [CSS inch](#), and thus if the image data is 300x600, it has [intrinsic dimensions](#) of 96 [CSS pixels](#) by 192 [CSS pixels](#).

All [img](#) and [link](#) elements are associated with a [source set](#).

A **source set** is an ordered set of zero or more [image sources](#) and a [source size](#).

An **image source** is a [URL](#), and optionally either a [pixel density descriptor](#), or a [width descriptor](#).

A **source size** is a [<source-size-value>](#). When a [source size](#) has a unit relative to the [viewport](#), it must be interpreted relative to the [img](#) element's [node document](#)'s [viewport](#). Other units must be interpreted the same as in Media Queries. [\[MQ\]](#)



---

A **parse error** for algorithms in this section indicates a non-fatal mismatch between input and requirements. User agents are encouraged to expose [parse errors](#) somehow.

---

Whether the image is fetched successfully or not (e.g. whether the response status was an [ok status](#)) must be ignored when determining the image's type and whether it is a valid image.

*This allows servers to return images with error responses, and have them displayed.*

The user agent should apply the [image sniffing rules](#) to determine the type of the image, with the image's [associated Content-Type headers](#) giving the *official type*. If these rules are not applied, then the type of the image must be the type given by the image's [associated Content-Type headers](#).

User agents must not support non-image resources with the [img](#) element (e.g. XML files whose [document element](#) is an HTML element). User agents must not run executable code (e.g. scripts) embedded in the image resource. User agents must only display the first page of a multipage resource (e.g. a PDF file). User agents must not allow the resource to act in an interactive fashion, but should honour any animation in the resource.

This specification does not specify which image types are to be supported.

#### **4.8.4.3.1 When to obtain images**

By default, images are obtained immediately. User agents may provide users with the option to instead obtain them on-demand. (The on-demand option might be used by bandwidth-constrained users, for example.)

When obtaining images immediately, the user agent must synchronously [update the image data](#) of the [img](#) element, with the *restart animation* flag set if so stated, whenever that element is created or has experienced [relevant mutations](#).

When obtaining images on demand, the user agent must [update the image data](#) of an [img](#) element whenever it needs the image data (i.e., on demand), but only if the [img](#) element's [current request](#)'s [state](#) is [unavailable](#). When an [img](#) element has

experienced [relevant mutations](#), if the user agent only obtains images on demand, the [img](#) element's [current request](#)'s [state](#) must return to [unavailable](#).

#### 4.8.4.3.2 *Reacting to DOM mutations*

The **relevant mutations** for an [img](#) element are as follows:

- The element's [src](#), [srcset](#), [width](#), or [sizes](#) attributes are set, changed, or removed.
- The element's [src](#) attribute is set to the same value as the previous value. This must set the *restart animation* flag for the [update the image data](#) algorithm.
- The element's [crossorigin](#) attribute's state is changed.
- The element's [referrerpolicy](#) attribute's state is changed.
- The [img](#) or [source](#) [HTML element insertion steps](#) or [HTML element removing steps](#) count the mutation as a [relevant mutation](#).
- The element's parent is a [picture](#) element and a [source](#) element that is a previous sibling has its [srcset](#), [sizes](#), [media](#), [type](#), [width](#) or [height](#) attributes set, changed, or removed.
- The element's [adopting steps](#) are run.

#### 4.8.4.3.3 *The list of available images*

Each [Document](#) object must have a **list of available images**. Each image in this list is identified by a tuple consisting of an [absolute URL](#), a [CORS settings attribute](#) mode, and, if the mode is not [No CORS](#), an [origin](#). Each image furthermore has an **ignore higher-layer caching** flag. User agents may copy entries from one [Document](#) object's [list of available images](#) to another at any time (e.g. when the [Document](#) is created, user agents can add to it all the images that are loaded in other [Documents](#)), but must not change the keys of entries copied in this way when doing so, and must unset the [ignore higher-layer caching](#) flag for the copied entry. User agents may also remove images from such lists at any time (e.g. to save memory). User agents must remove entries in the [list of available images](#) as appropriate given higher-layer caching semantics for the resource (e.g. the HTTP [`Cache-Control`](#) response header) when the [ignore higher-layer caching](#) flag is unset.

The [list of available images](#) is intended to enable synchronous switching when changing the `src` attribute to a URL that has previously been loaded, and to avoid re-downloading images in the same document even when they don't allow caching per HTTP. It is not used to avoid re-downloading the same image while the previous image is still loading.

The user agent can also store the image data separately from the [list of available images](#).

For example, if a resource has the HTTP response header `Cache-Control: must-revalidate`, and its [ignore higher-layer caching](#) flag is unset, the user agent would remove it from the [list of available images](#) but could keep the image data separately, and use that if the server responds with a 304 Not Modified status.

#### 4.8.4.3.4 Decoding images

Image data is usually encoded in order to reduce file size. This means that in order for the user agent to present the image to the screen, the data needs to be decoded. **Decoding** is the process which converts an image's media data into a bitmap form, suitable for presentation to the screen. Note that this process can be slow relative to other processes involved in presenting content. Thus, the user agent can choose when to perform decoding, in order to create the best user experience.

Image decoding is said to be synchronous if it prevents presentation of other content until it is finished. Typically, this has an effect of atomically presenting the image and any other content at the same time. However, this presentation is delayed by the amount of time it takes to perform the decode.

Image decoding is said to be asynchronous if it does not prevent presentation of other content. This has an effect of presenting non-image content faster. However, the image content is missing on screen until the decode finishes. Once the decode is finished, the screen is updated with the image.

In both synchronous and asynchronous decoding modes, the final content is presented to screen after the same amount of time has elapsed. The main difference is whether the user agent presents non-image content ahead of presenting the final content.

In order to aid the user agent in deciding whether to perform synchronous or asynchronous decode, the [decoding](#) attribute can be set on `img` elements. The possible values of the [decoding](#) attribute are the following **image decoding hint** keywords:

Keyword	State	Description
<code>sync</code>	<b>Sync</b>	Indicates a preference to <a href="#">decode</a> this image synchronously for atomic presentation with other content.

Keyword	State	Description
<b>async</b>	<b>Async</b>	Indicates a preference to <a href="#">decode</a> this image asynchronously to avoid delaying presentation of other content.
<b>auto</b>	<b>Auto</b>	Indicates no preference in decoding mode (the default).

When [decoding](#) an image, the user agent should respect the preference indicated by the [decoding](#) attribute's state. If the state indicated is [auto](#), then the user agent is free to choose any decoding behavior.

*It is also possible to control the decoding behavior using the [decode\(\)](#) method. Since the [decode\(\)](#) method performs [decoding](#) independently from the process responsible for presenting content to screen, it is unaffected by the [decoding](#) attribute.*

#### 4.8.4.3.5 Updating the image data

*This algorithm cannot be called from steps running [in parallel](#). If a user agent needs to call this algorithm from steps running [in parallel](#), it needs to [queue](#) a task to do so.*

When the user agent is to **update the image data** of an [img](#) element, optionally with the *restart animations* flag set, it must run the following steps:

1. If the element's [node document](#) is not [fully active](#), then:
  1. Continue running this algorithm [in parallel](#).
  2. Wait until the element's [node document](#) is [fully active](#).
  3. If another instance of this algorithm for this [img](#) element was started after this instance (even if it aborted and is no longer running), then return.
  4. [Queue a microtask](#) to continue this algorithm.
2. If the user agent cannot support images, or its support for images has been disabled, then [abort the image request](#) for the [current request](#) and the [pending request](#), set [current request](#)'s [state](#) to [unavailable](#), set [pending request](#) to null, and return.
3. Let *selected source* be null and *selected pixel density* be undefined.
4. If the element does not [use srcset or picture](#) and it has a [src](#) attribute specified whose value is not the empty string, then set *selected source* to the value of the element's [src](#) attribute and set *selected pixel density* to 1.0.
5. Set the element's [last selected source](#) to *selected source*.

6. If *selected source* is not null, then:
  1. [Parse](#) *selected source*, relative to the element's [node document](#). If that is not successful, then abort this inner set of steps. Otherwise, let *urlString* be the [resulting URL string](#).
  2. Let *key* be a tuple consisting of *urlString*, the [img](#) element's [crossorigin](#) attribute's mode, and, if that mode is not [No CORS](#), the [node document](#)'s [origin](#).
  3. If the [list of available images](#) contains an entry for *key*, then:
    1. Set the [ignore higher-layer caching](#) flag for that entry.
    2. [Abort the image request](#) for the [current request](#) and the [pending request](#).
    3. Set [pending request](#) to null.
    4. Let [current request](#) be a new [image request](#) whose [image data](#) is that of the entry and whose [state](#) is [completely available](#).
    5. [Prepare current request for presentation](#) given *img*.
    6. Set [current request](#)'s [current pixel density](#) to *selected pixel density*.
    7. [Queue an element task](#) on the [DOM manipulation task source](#) given the [img](#) element and following steps:
      1. If *restart animation* is set, then [restart the animation](#).
      2. Set [current request](#)'s [current URL](#) to *urlString*.
      3. [Fire an event](#) named [load](#) at the [img](#) element.
    8. Abort the [update the image data](#) algorithm.
  7. [Queue a microtask](#) to perform the rest of this algorithm, allowing the [task](#) that invoked this algorithm to continue.
  8. If another instance of this algorithm for this [img](#) element was started after this instance (even if it aborted and is no longer running), then return.

*Only the last instance takes effect, to avoid multiple requests when, for example, the [src](#), [srcset](#), and [crossorigin](#) attributes are all set in succession.*
  9. Let *selected source* and *selected pixel density* be the URL and pixel density that results from [selecting an image source](#), respectively.
  10. If *selected source* is null, then:

1. Set the [current request](#)'s [state](#) to [broken](#), [abort the image request](#) for the [current request](#) and the [pending request](#), and set [pending request](#) to null.
  2. [Queue an element task](#) on the [DOM manipulation task source](#) given the [img](#) element and the following steps:
    1. Change the [current request](#)'s [current URL](#) to the empty string.
    2. If the element has a [src](#) attribute or it [uses srcset or picture](#), [fire an event](#) named [error](#) at the [img](#) element.
  3. Return.
11. [Parse](#) *selected source*, relative to the element's [node document](#), and let *urlString* be the [resulting URL string](#). If that is not successful, then:
1. [Abort the image request](#) for the [current request](#) and the [pending request](#).
  2. Set the [current request](#)'s [state](#) to [broken](#).
  3. Set [pending request](#) to null.
  4. [Queue an element task](#) on the [DOM manipulation task source](#) given the [img](#) element and the following steps:
    1. Change the [current request](#)'s [current URL](#) to *selected source*.
    2. [Fire an event](#) named [error](#) at the [img](#) element.
  5. Return.
12. If the [pending request](#) is not null and *urlString* is the same as the [pending request](#)'s [current URL](#), then return.
13. If *urlString* is the same as the [current request](#)'s [current URL](#) and [current request](#)'s [state](#) is [partially available](#), then [abort the image request](#) for the [pending request](#), [queue an element task](#) on the [DOM manipulation task source](#) given the [img](#) element to [restart the animation](#) if *restart animation* is set, and return.
14. If the [pending request](#) is not null, then [abort the image request](#) for the [pending request](#).
15. Set *image request* to a new [image request](#) whose [current URL](#) is *urlString*.
16. If [current request](#)'s [state](#) is [unavailable](#) or [broken](#), then set the [current request](#) to *image request*. Otherwise, set the [pending request](#) to *image request*.

17. Let *request* be the result of [creating a potential-CORS request](#) given *urlString*, "image", and the current state of the element's [crossorigin](#) content attribute.
18. Set *request*'s [client](#) to the element's [node document](#)'s [relevant settings object](#).
19. If the element [uses srcset or picture](#), set *request*'s [initiator](#) to "imageset".
20. Set *request*'s [referrer policy](#) to the current state of the element's [referrerpolicy](#) attribute.
21. Let *delay load event* be true if the [img](#)'s [lazy loading attribute](#) is in the [Eager](#) state, or if [scripting is disabled](#) for the [img](#), and false otherwise.
22. If the [will lazy load element steps](#) given the [img](#) return true, then:
  1. Set the [img](#)'s [lazy load resumption steps](#) to the rest of this algorithm starting with the step labeled *fetch the image*.
  2. [Start intersection-observing a lazy loading element](#) for the [img](#) element.
  3. Return.
23. *Fetch the image*: [Fetch](#) *request*. Return from this algorithm, and run the remaining steps as part of the fetch's [processResponse](#) for the [response](#) response.

The resource obtained in this fashion, if any, is *image request*'s [image data](#). It can be either [CORS-same-origin](#) or [CORS-cross-origin](#); this affects the image's interaction with other APIs (e.g., when used on a [canvas](#)).

When *delay load event* is true, fetching the image must [delay the load event](#) of the element's [node document](#) until the [task](#) that is [queued](#) by the [networking task source](#) once the resource has been fetched ([defined below](#)) has been run.

***This, unfortunately, can be used to perform a rudimentary port scan of the user's local network (especially in conjunction with scripting, though scripting isn't actually necessary to carry out such an attack). User agents may implement [cross-origin](#) access control policies that are stricter than those described above to mitigate this attack, but unfortunately such policies are typically not compatible with existing web content.***

24. As soon as possible, jump to the first applicable entry from the following list:

**If the resource type is [multipart/x-mixed-replace](#)**

The next [task](#) that is [queued](#) by the [networking task source](#) while the image is being fetched must run the following steps:



1. If *image request* is the [pending request](#) and at least one body part has been completely decoded, [abort the image request](#) for the [current request](#), [upgrade the pending request to the current request](#).
2. Otherwise, if *image request* is the [pending request](#) and the user agent is able to determine that *image request*'s image is corrupted in some fatal way such that the image dimensions cannot be obtained, [abort the image request](#) for the [current request](#), [upgrade the pending request to the current request](#), and set the [current request](#)'s [state](#) to [broken](#).
3. Otherwise, if *image request* is the [current request](#), its [state](#) is [unavailable](#), and the user agent is able to determine *image request*'s image's width and height, set the [current request](#)'s [state](#) to [partially available](#).
4. Otherwise, if *image request* is the [current request](#), its [state](#) is [unavailable](#), and the user agent is able to determine that *image request*'s image is corrupted in some fatal way such that the image dimensions cannot be obtained, set the [current request](#)'s [state](#) to [broken](#).

Each [task](#) that is [queued](#) by the [networking task source](#) while the image is being fetched must update the presentation of the image, but as each new body part comes in, if the user agent is able to determine the image's width and height, it must [prepare the \[img\]\(#\) element's current request for presentation](#) given the [img](#) element and replace the previous image. Once one body part has been completely decoded, perform the following steps:

5. Set the [img](#) element's [current request](#)'s [state](#) to [completely available](#).
6. [Queue an element task](#) on the [DOM manipulation task source](#) given the [img](#) element to [fire an event](#) named [load](#) at the [img](#) element.

**If the resource type and data corresponds to a supported image format, [as described below](#)**

The next [task](#) that is [queued](#) by the [networking task source](#) while the image is being fetched must run the following steps:

7. If the user agent is able to determine *image request*'s image's width and height, and *image request* is [pending request](#), set *image request*'s [state](#) to [partially available](#).
8. Otherwise, if the user agent is able to determine *image request*'s image's width and height, and *image request* is [current request](#), [prepare \*image request\* for presentation](#) given the [img](#) element and set *image request*'s [state](#) to [partially available](#).
9. Otherwise, if the user agent is able to determine that *image request*'s image is corrupted in some fatal way such that the image dimensions cannot be obtained, and *image request* is [pending request](#):



1. [Abort the image request](#) for the [current request](#) and the [pending request](#).
  2. [Upgrade the pending request to the current request](#).
  3. Set [current request](#)'s [state](#) to [broken](#).
  4. [Fire an event](#) named [error](#) at the [img](#) element.
10. Otherwise, if the user agent is able to determine that *image request*'s image is corrupted in some fatal way such that the image dimensions cannot be obtained, and *image request* is [current request](#):
1. [Abort the image request](#) for *image request*.
  2. [Fire an event](#) named [error](#) at the [img](#) element.

That [task](#), and each subsequent [task](#), that is [queued](#) by the [networking task source](#) while the image is being fetched, if *image request* is the [current request](#), must update the presentation of the image appropriately (e.g., if the image is a progressive JPEG, each packet can improve the resolution of the image).

Furthermore, the last [task](#) that is [queued](#) by the [networking task source](#) once the resource has been fetched must additionally run these steps:

11. If *image request* is the [pending request](#), [abort the image request](#) for the [current request](#), [upgrade the pending request to the current request](#) and [prepare image request for presentation](#) given the [img](#) element.
12. Set *image request* to the [completely available](#) state.
13. Add the image to the [list of available images](#) using the key *key*, with the [ignore higher-layer caching](#) flag set.
14. [Fire an event](#) named [load](#) at the [img](#) element.

### Otherwise

The image data is not in a supported file format; the user agent must set *image request*'s [state](#) to [broken](#), [abort the image request](#) for the [current request](#) and the [pending request](#), [upgrade the pending request to the current request](#) if *image request* is the [pending request](#), and then [queue an element task](#) on the [DOM manipulation task source](#) given the [img](#) element to [fire an event](#) named [error](#) at the [img](#) element.

While a user agent is running the above algorithm for an element *x*, there must be a strong reference from the element's [node document](#) to the element *x*, even if that element is not [connected](#).

To **abort the image request** for an [image request](#) *image request* means to run the following steps:

1. Forget *image request*'s [image data](#), if any.
2. Abort any instance of the [fetching](#) algorithm for *image request*, discarding any pending tasks generated by that algorithm.

To **upgrade the pending request to the current request** for an [img](#) element means to run the following steps:

1. Let the [img](#) element's [current request](#) be the [pending request](#).
2. Let the [img](#) element's [pending request](#) be null.

#### 4.8.4.3.6 Preparing an image for presentation

To **prepare an image for presentation** for an [image request](#) *req* given image element *img*:

1. Let *exifTagMap* be the EXIF tags obtained from *req*'s [image data](#), as defined by the relevant codec. [\[EXIF\]](#)
2. Let *physicalWidth* and *physicalHeight* be the width and height obtained from *req*'s [image data](#), as defined by the relevant codec.
3. Let *dimX* be the value of *exifTagMap*'s tag 0xA002 ([PixelXDimension](#)).
4. Let *dimY* be the value of *exifTagMap*'s tag 0xA003 ([PixelYDimension](#)).
5. Let *resX* be the value of *exifTagMap*'s tag 0x011A ([XResolution](#)).
6. Let *resY* be the value of *exifTagMap*'s tag 0x011B ([YResolution](#)).
7. Let *resUnit* be the value of *exifTagMap*'s tag 0x0128 ([ResolutionUnit](#)).
8. If either *dimX* or *dimY* is not a positive integer, then return.
9. If either *resX* or *resY* is not a positive floating-point number, then return.
10. If *resUnit* is not equal to 2 ([Inch](#)), then return.
11. Let *widthFromDensity* be the value of *physicalWidth*, multiplied by 72 and divided by *resX*.
12. Let *heightFromDensity* be the value of *physicalHeight*, multiplied by 72 and divided by *resY*.

13. If *widthFromDensity* is not equal to *dimX* or *heightFromDensity* is not equal to *dimY*, then return.
14. If *req*'s [image data](#) is [CORS-cross-origin](#), then set *img*'s [intrinsic dimensions](#) to *dimX* and *dimY*, scale *img*'s pixel data accordingly, and return.
15. Set *req*'s [preferred density-corrected dimensions](#) to a struct with its width set to *dimX* and its height set to *dimY*.
16. Update *req*'s [img](#) element's presentation appropriately.

*Resolution in EXIF is equivalent to [CSS points per inch](#), therefore 72 is the base for computing size from resolution.*

It is not yet specified what would be the case if EXIF arrives after the image is already presented. See [issue #4929](#).

#### **4.8.4.3.7 Selecting an image source**

To **select an image source** given an [img](#) element *e*:

1. [Update the source set](#) for *e*.
2. If *e*'s [source set](#) is empty, return null as the URL and undefined as the pixel density.
3. Return the result of [selecting an image](#) from *e*'s [source set](#).

To **select an image source from a source set** given a [source set](#) *sourceSet*:

1. If an entry *b* in *sourceSet* has the same associated [pixel density descriptor](#) as an earlier entry *a* in *sourceSet*, then remove entry *b*. Repeat this step until none of the entries in *sourceSet* have the same associated [pixel density descriptor](#) as an earlier entry.
2. In an [implementation-defined](#) manner, choose one [image source](#) from *sourceSet*. Let this be *selectedSource*.
3. Return *selectedSource* and its associated pixel density.

#### **4.8.4.3.8 Creating a source set from attributes**

When asked to **create a source set** given a string *default source*, a string *srcset* and a string *sizes*:

1. Let *source set* be an empty [source set](#).
2. If *srcset* is not an empty string, then set *source set* to the result of [parsing](#) *srcset*.
3. Let [source size](#) be the result of [parsing](#) *sizes*.
4. If *default source* is not the empty string and *source set* does not contain an [image source](#) with a [pixel density descriptor](#) value of 1, and no [image source](#) with a [width descriptor](#), append *default source* to *source set*.
5. [Normalize the source densities](#) of *source set*.
6. Return *source set*.

#### 4.8.4.3.9 Updating the source set

When asked to **update the source set** for a given [img](#) or [link](#) element *el*, user agents must do the following:

1. Set *el*'s [source set](#) to an empty [source set](#).
2. Let *elements* be « *el* ».
3. If *el* is an [img](#) element whose parent node is a [picture](#) element, then [replace](#) the contents of *elements* with *el*'s parent node's child elements, retaining relative order.
4. [For each](#) *child* in *elements*:
  1. If *child* is *el*:
    1. Let *default source* be the empty string.
    2. Let *srcset* be the empty string.
    3. Let *sizes* be the empty string.
    4. If *el* is an [img](#) element that has a [srcset](#) attribute, then set *srcset* to that attribute's value.
    5. Otherwise, if *el* is a [link](#) element that has an [imagesrcset](#) attribute, then set *srcset* to that attribute's value.
    6. If *el* is an [img](#) element that has a [sizes](#) attribute, then set *sizes* to that attribute's value.

7. Otherwise, if *e* is a [link](#) element that has an [imagesizes](#) attribute, then set *sizes* to that attribute's value.
8. If *e* is an [img](#) element that has a [src](#) attribute, then set *default source* to that attribute's value.
9. Otherwise, if *e* is a [link](#) element that has an [href](#) attribute, then set *default source* to that attribute's value.
10. Let *e*'s [source set](#) be the result of [creating a source set](#) given *default source*, *srcset*, and *sizes*.
11. Return.

*If *e* is a [link](#) element, then *elements* contains only *e*, so this step will be reached immediately and the rest of the algorithm will not run.*

2. If *child* is not a [source](#) element, then [continue](#).
3. If *child* does not have a [srcset](#) attribute, [continue](#) to the next child.
4. [Parse \*child\*'s \*srcset\* attribute](#) and let the returned [source set](#) be *source set*.
5. If *source set* has zero [image sources](#), [continue](#) to the next child.
6. If *child* has a [media](#) attribute, and its value does not [match the environment](#), [continue](#) to the next child.
7. [Parse \*child\*'s \*sizes\* attribute](#), and let *source set*'s [source size](#) be the returned value.
8. If *child* has a [type](#) attribute, and its value is an unknown or unsupported [MIME type](#), [continue](#) to the next child.
9. If *child* has [width](#) or [height](#) attributes, set *e*'s [dimension attribute source](#) to *child*. Otherwise, set *e*'s [dimension attribute source](#) to *e*.
10. [Normalize the source densities](#) of *source set*.
11. Let *e*'s [source set](#) be *source set*.
12. Return.

*Each [img](#) element independently considers its previous sibling [source](#) elements plus the [img](#) element itself for selecting an [image source](#), ignoring any other (invalid) elements, including other [img](#) elements in the same [picture](#) element, or [source](#) elements that are following siblings of the relevant [img](#) element.*

#### 4.8.4.3.10 Parsing a srcset attribute

When asked to **parse a srcset attribute** from an element, parse the value of the element's [srcset attribute](#) as follows:

1. Let *input* be the value passed to this algorithm.
2. Let *position* be a pointer into *input*, initially pointing at the start of the string.
3. Let *candidates* be an initially empty [source set](#).
4. *Splitting loop*: [Collect a sequence of code points](#) that are [ASCII whitespace](#) or U+002C COMMA characters from *input* given *position*. If any U+002C COMMA characters were collected, that is a [parse error](#).
5. If *position* is past the end of *input*, return *candidates*.
6. [Collect a sequence of code points](#) that are not [ASCII whitespace](#) from *input* given *position*, and let that be *url*.
7. Let *descriptors* be a new empty list.
8. If *url* ends with U+002C (,), then:
  1. Remove all trailing U+002C COMMA characters from *url*. If this removed more than one character, that is a [parse error](#).

Otherwise:

2. *Descriptor tokenizer*: [Skip ASCII whitespace](#) within *input* given *position*.
3. Let *current descriptor* be the empty string.
4. Let *state* be *in descriptor*.
5. Let *c* be the character at *position*. Do the following depending on the value of *state*. For the purpose of this step, "EOF" is a special character representing that *position* is past the end of *input*.

##### ***In descriptor***

Do the following, depending on the value of *c*:

##### **[ASCII whitespace](#)**

If *current descriptor* is not empty, append *current descriptor* to *descriptors* and let *current descriptor* be the empty string. Set *state* to *after descriptor*.

##### **U+002C COMMA (,)**

Advance *position* to the next character in *input*. If *current descriptor* is not empty, append *current descriptor* to *descriptors*. Jump to the step labeled *descriptor parser*.

**U+0028 LEFT PARENTHESIS (()**

Append *c* to *current descriptor*. Set *state* to *in parens*.

**EOF**

If *current descriptor* is not empty, append *current descriptor* to *descriptors*. Jump to the step labeled *descriptor parser*.

**Anything else**

Append *c* to *current descriptor*.

***In parens***

Do the following, depending on the value of *c*:

**U+0029 RIGHT PARENTHESIS ())**

Append *c* to *current descriptor*. Set *state* to *in descriptor*.

**EOF**

Append *current descriptor* to *descriptors*. Jump to the step labeled *descriptor parser*.

**Anything else**

Append *c* to *current descriptor*.

***After descriptor***

Do the following, depending on the value of *c*:

**ASCII whitespace**

Stay in this state.

**EOF**

Jump to the step labeled *descriptor parser*.

**Anything else**

Set *state* to *in descriptor*. Set *position* to the *previous* character in *input*.

Advance *position* to the next character in *input*. Repeat this step.

*In order to be compatible with future additions, this algorithm supports multiple descriptors and descriptors with parens.*

9. *Descriptor parser*. Let *error* be *no*.

10. Let *width* be *absent*.

11. Let *density* be *absent*.

12. Let *future-compat-h* be *absent*.

13. For each descriptor in *descriptors*, run the appropriate set of steps from the following list:

**If the descriptor consists of a [valid non-negative integer](#) followed by a U+0077 LATIN SMALL LETTER W character**

1. If the user agent does not support the [sizes](#) attribute, let *error* be yes.

*A conforming user agent will support the [sizes](#) attribute. However, user agents typically implement and ship features in an incremental manner in practice.*

2. If *width* and *density* are not both *absent*, then let *error* be yes.
3. Apply the [rules for parsing non-negative integers](#) to the descriptor. If the result is zero, let *error* be yes. Otherwise, let *width* be the result.

**If the descriptor consists of a [valid floating-point number](#) followed by a U+0078 LATIN SMALL LETTER X character**

4. If *width*, *density* and *future-compat-h* are not all *absent*, then let *error* be yes.
5. Apply the [rules for parsing floating-point number values](#) to the descriptor. If the result is less than zero, let *error* be yes. Otherwise, let *density* be the result.

*If density is zero, the [intrinsic dimensions](#) will be infinite. User agents are [expected to have limits](#) in how big images can be rendered.*

**If the descriptor consists of a [valid non-negative integer](#) followed by a U+0068 LATIN SMALL LETTER H character**

This is a [parse error](#).

6. If *future-compat-h* and *density* are not both *absent*, then let *error* be yes.
7. Apply the [rules for parsing non-negative integers](#) to the descriptor. If the result is zero, let *error* be yes. Otherwise, let *future-compat-h* be the result.

### **Anything else**

Let *error* be yes.

14. If *future-compat-h* is not *absent* and *width* is *absent*, let *error* be yes.

15. If *error* is still *no*, then append a new [image source](#) to *candidates* whose URL is *url*, associated with a width *width* if not *absent* and a pixel density *density* if not *absent*. Otherwise, there is a [parse error](#).

16. Return to the step labeled *splitting loop*.



#### 4.8.4.3.11 Parsing a sizes attribute

When asked to **parse a sizes attribute** from an element, [parse a comma-separated list of component values](#) from the value of the element's [sizes attribute](#) (or the empty string, if the attribute is absent), and let *unparsed sizes list* be the result. [\[CSSSYNTAX\]](#)

For each *unparsed size* in *unparsed sizes list*:

1. Remove all consecutive [<whitespace-token>](#)s from the end of *unparsed size*. If *unparsed size* is now empty, that is a [parse error](#); [continue](#).
2. If the last [component value](#) in *unparsed size* is a valid non-negative [<source-size-value>](#), let *size* be its value and remove the [component value](#) from *unparsed size*. Any CSS function other than the [math functions](#) is invalid. Otherwise, there is a [parse error](#); [continue](#).
3. Remove all consecutive [<whitespace-token>](#)s from the end of *unparsed size*. If *unparsed size* is now empty, return *size* and exit this algorithm. If this was not the last item in *unparsed sizes list*, that is a [parse error](#).
4. Parse the remaining [component values](#) in *unparsed size* as a [<media-condition>](#). If it does not parse correctly, or it does parse correctly but the [<media-condition>](#) evaluates to false, [continue](#). [\[MQ\]](#)
5. Return *size* and exit this algorithm.

If the above algorithm exhausts *unparsed sizes list* without returning a size value, then return 100vw.

*While a [valid source size list](#) only contains a bare [<source-size-value>](#) (without an accompanying [<media-condition>](#)) as the last entry in the [<source-size-list>](#), the parsing algorithm technically allows such at any point in the list, and will accept it immediately as the size if the preceding entries in the list weren't used. This is to enable future extensions, and protect against simple author errors such as a final trailing comma.*

#### 4.8.4.3.12 Normalizing the source densities

An [image source](#) can have a [pixel density descriptor](#), a [width descriptor](#), or no descriptor at all accompanying its URL. Normalizing a [source set](#) gives every [image source](#) a [pixel density descriptor](#).

When asked to **normalize the source densities** of a [source set](#) source set, the user agent must do the following:

1. Let *source size* be source set's [source size](#).

2. For each [image source](#) in *source set*:
  1. If the [image source](#) has a [pixel density descriptor](#), [continue](#) to the next [image source](#).
  2. Otherwise, if the [image source](#) has a [width descriptor](#), replace the [width descriptor](#) with a [pixel density descriptor](#) with a [value](#) of the [width descriptor value](#) divided by the [source size](#) and a unit of  $\times$ .
 

*If the [source size](#) is zero, the density would be infinity, which results in the [intrinsic dimensions](#) being zero by zero.*
  3. Otherwise, give the [image source](#) a [pixel density descriptor](#) of  $1\times$ .

#### 4.8.4.3.13 Reacting to environment changes

The user agent may at any time run the following algorithm to update an [img](#) element's image in order to react to changes in the environment. (User agents are *not required* to ever run this algorithm; for example, if the user is not looking at the page any more, the user agent might want to wait until the user has returned to the page before determining which image to use, in case the environment changes again in the meantime.)

*User agents are encouraged to run this algorithm in particular when the user changes the [viewport](#)'s size (e.g. by resizing the window or changing the page zoom), and when an [img](#) element is [inserted into a document](#), so that the [density-corrected intrinsic width and height](#) match the new [viewport](#), and so that the correct image is chosen when [art direction](#) is involved.*

1. [Await a stable state](#). The [synchronous section](#) consists of all the remaining steps of this algorithm until the algorithm says the [synchronous section](#) has ended. (Steps in [synchronous sections](#) are marked with ⌚.)
2. ⌚ If the [img](#) element does not [use srcset or picture](#), its [node document](#) is not [fully active](#), has image data whose resource type is [multipart/x-mixed-replace](#), or the [pending request](#) is not null, then return.
3. ⌚ Let *selected source* and *selected pixel density* be the URL and pixel density that results from [selecting an image source](#), respectively.
4. ⌚ If *selected source* is null, then return.
5. ⌚ If *selected source* and *selected pixel density* are the same as the element's [last selected source](#) and [current pixel density](#), then return.
6. ⌚ [Parse](#) *selected source*, relative to the element's [node document](#), and let *urlString* be the [resulting URL string](#). If that is not successful, then return.

7. ⌚ Let *corsAttributeState* be the state of the element's [crossorigin](#) content attribute.
8. ⌚ Let *origin* be the [img](#) element's [node document](#)'s [origin](#).
9. ⌚ Let *client* be the [img](#) element's [node document](#)'s [relevant settings object](#).
10. ⌚ Let *key* be a tuple consisting of *urlString*, *corsAttributeState*, and, if *corsAttributeState* is not [No CORS](#), *origin*.
11. ⌚ Let *image request* be a new [image request](#) whose [current URL](#) is *urlString*.
12. ⌚ Let the element's [pending request](#) be *image request*.
13. End the [synchronous section](#), continuing the remaining steps [in parallel](#).
14. If the [list of available images](#) contains an entry for *key*, then set *image request*'s [image data](#) to that of the entry. Continue to the next step.

Otherwise:

1. Let *request* be the result of [creating a potential-CORS request](#) given *urlString*, "image", and *corsAttributeState*.
  2. Set *request*'s [client](#) to *client*, [initiator](#) to "imageset", and set *request*'s [synchronous flag](#).
  3. Set *request*'s [referrer policy](#) to the current state of the element's [referrerpolicy](#) attribute.
  4. Let *response* be the result of [fetching request](#).
  5. If *response*'s [unsafe response](#) is a [network error](#) or if the image format is unsupported (as determined by applying the [image sniffing rules](#), again as mentioned earlier), or if the user agent is able to determine that *image request*'s image is corrupted in some fatal way such that the image dimensions cannot be obtained, or if the resource type is [multipart/x-mixed-replace](#), then let [pending request](#) be null and abort these steps.
  6. Otherwise, *response*'s [unsafe response](#) is *image request*'s [image data](#). It can be either [CORS-same-origin](#) or [CORS-cross-origin](#); this affects the image's interaction with other APIs (e.g., when used on a [canvas](#)).
15. [Queue an element task](#) on the [DOM manipulation task source](#) given the [img](#) element and the following steps:
1. If the [img](#) element has experienced [relevant mutations](#) since this algorithm started, then let [pending request](#) be null and abort these steps.

2. Let the `img` element's [last selected source](#) be *selected source* and the `img` element's [current pixel density](#) be *selected pixel density*.
3. Set the *image request's* [state](#) to [completely available](#).
4. Add the image to the [list of available images](#) using the key *key*, with the [ignore higher-layer caching](#) flag set.
5. [Upgrade the pending request to the current request](#).
6. [Prepare image request for presentation](#) given the `img` element.
7. [Fire an event](#) named [load](#) at the `img` element.

#### 4.8.4.4 Requirements for providing text to act as an alternative for images

##### 4.8.4.4.1 General guidelines

Except where otherwise specified, the [alt](#) attribute must be specified and its value must not be empty; the value must be an appropriate replacement for the image. The specific requirements for the [alt](#) attribute depend on what the image is intended to represent, as described in the following sections.

The most general rule to consider when writing alternative text is the following: **the intent is that replacing every image with the text of its [alt](#) attribute does not change the meaning of the page.**

So, in general, alternative text can be written by considering what one would have written had one not been able to include the image.

A corollary to this is that the [alt](#) attribute's value should never contain text that could be considered the image's *caption*, *title*, or *legend*. It is supposed to contain replacement text that could be used by users *instead* of the image; it is not meant to supplement the image. The [title](#) attribute can be used for supplemental information.

Another corollary is that the [alt](#) attribute's value should not repeat information that is already provided in the prose next to the image.

*One way to think of alternative text is to think about how you would read the page containing the image to someone over the phone, without mentioning that there is an image present. Whatever you say instead of the image is typically a good start for writing the alternative text.*

#### 4.8.4.4.2 A link or button containing nothing but the image

When an `a` element that creates a [hyperlink](#), or a `button` element, has no textual content but contains one or more images, the `alt` attributes must contain text that together convey the purpose of the link or button.

In this example, a user is asked to pick their preferred color from a list of three. Each color is given by an image, but for users who have configured their user agent not to display images, the color names are used instead:

```
<h1>Pick your color</h1>
<ul>
  <li><a href="green.html"></a></li>
  <li><a href="blue.html"></a></li>
  <li><a href="red.html"></a></li>
</ul>
```

In this example, each button has a set of images to indicate the kind of color output desired by the user. The first image is used in each case to give the alternative text.

```
<button name="rgb"></button>
<button name="cmym"></button>
```

Since each image represents one part of the text, it could also be written like this:

```
<button name="rgb"></button>
<button name="cmym"></button>
```

However, with other alternative text, this might not work, and putting all the alternative text into one image in each case might make more sense:

```
<button name="rgb"></button>
<button name="cmym"></button>
```

#### 4.8.4.4.3 A phrase or paragraph with an alternative graphical representation: charts, diagrams, graphs, maps, illustrations

Sometimes something can be more clearly stated in graphical form, for example as a flowchart, a diagram, a graph, or a simple map showing directions. In such cases, an image can be given using the `img` element, but the lesser textual version must still be given, so that users who are unable to view the image (e.g. because they have a very slow connection, or because they are using a text-only browser, or because they are listening to the page being read out by a hands-free automobile voice web browser, or simply because they are blind) are still able to understand the message being conveyed.

The text must be given in the `alt` attribute, and must convey the same message as the image specified in the `src` attribute.

It is important to realize that the alternative text is a *replacement* for the image, not a description of the image.

In the following example we have [a flowchart](#) in image form, with text in the `alt` attribute rephrasing the flowchart in prose form:

```
<p>In the common case, the data handled by the tokenization stage
comes from the network, but it can also come from script.</p>
<p></p>
```

Here's another example, showing a good solution and a bad solution to the problem of including an image in a description.

First, here's the good solution. This sample shows how the alternative text should just be what you would have put in the prose if the image had never existed.

```
<!-- This is the correct way to do things. -->
<p>
  You are standing in an open field west of a house.

  There is a small mailbox here.
</p>
```

Second, here's the bad solution. In this incorrect way of doing things, the alternative text is simply a description of the image, instead of a textual replacement for the image. It's bad because when the image isn't shown, the text doesn't flow as well as in the first example.

```

<!-- This is the wrong way to do things. -->
<p>
  You are standing in an open field west of a house.

  There is a small mailbox here.
</p>

```

Text such as "Photo of white house with boarded door" would be equally bad alternative text (though it could be suitable for the [title](#) attribute or in the [figcaption](#) element of a [figure](#) with this image).

#### 4.8.4.4.4 A short phrase or label with an alternative graphical representation: icons, logos

A document can contain information in iconic form. The icon is intended to help users of visual browsers to recognize features at a glance.

In some cases, the icon is supplemental to a text label conveying the same meaning. In those cases, the [alt](#) attribute must be present but must be empty.

Here the icons are next to text that conveys the same meaning, so they have an empty [alt](#) attribute:

```

<nav>
  <p><a href="/help/">
  Help</a></p>
  <p><a href="/configure/">
  Configuration Tools</a></p>
</nav>

```

In other cases, the icon has no text next to it describing what it means; the icon is supposed to be self-explanatory. In those cases, an equivalent textual label must be given in the [alt](#) attribute.

Here, posts on a news site are labeled with an icon indicating their topic.

```

<body>
  <article>
    <header>
      <h1>Ratatouille wins <i>Best Movie of the Year</i> award</h1>
      <p></p>
    </header>

```

```

<p>Pixar has won yet another <i>Best Movie of the Year</i> award,
making this its 8th win in the last 12 years.</p>
</article>
<article>
  <header>
    <h1>Latest TWiT episode is online</h1>
    <p></p>
  </header>
  <p>The latest TWiT episode has been posted, in which we hear
several tech news stories as well as learning much more about the
iPhone. This week, the panelists compare how reflective their
iPhones' Apple logos are.</p>
</article>
</body>

```

Many pages include logos, insignia, flags, or emblems, which stand for a particular entity such as a company, organization, project, band, software package, country, or some such.

If the logo is being used to represent the entity, e.g. as a page heading, the alt attribute must contain the name of the entity being represented by the logo. The alt attribute must *not* contain text like the word "logo", as it is not the fact that it is a logo that is being conveyed, it's the entity itself.

If the logo is being used next to the name of the entity that it represents, then the logo is supplemental, and its alt attribute must instead be empty.

If the logo is merely used as decorative material (as branding, or, for example, as a side image in an article that mentions the entity to which the logo belongs), then the entry below on purely decorative images applies. If the logo is actually being discussed, then it is being used as a phrase or paragraph (the description of the logo) with an alternative graphical representation (the logo itself), and the first entry above applies.

In the following snippets, all four of the above cases are present. First, we see a logo used to represent a company:

```

<h1></h1>

```

Next, we see a paragraph which uses a logo right next to the company name, and so doesn't have any alternative text:

```

<article>
  <h2>News</h2>
  <p>We have recently been looking at buying the <img
src="alpha.gif"

```



```
alt=""> ABT company, a small Greek company  
specializing in our type of product.</p>
```

In this third snippet, we have a logo being used in an aside, as part of the larger article discussing the acquisition:

```
<aside><p></p></aside>  
  
<p>The ABT company has had a good quarter, and our  
pie chart studies of their accounts suggest a much bigger blue  
slice  
than its green and orange slices, which is always a good sign.</p>  
</article>
```

Finally, we have an opinion piece talking about a logo, and the logo is therefore described in detail in the alternative text.

```
<p>Consider for a moment their logo:</p>  
  
<p></p>  
  
<p>How unoriginal can you get? I mean, ooooooh, a question mark, how  
<em>revolutionary</em>, how utterly <em>ground-breaking</em>, I'm  
sure everyone will rush to adopt those specifications now! They  
could  
at least have tried for some sort of, I don't know, sequence of  
rounded squares with varying shades of green and bold white  
outlines,  
at least that would look good on the cover of a blue book.</p>
```

This example shows how the alternative text should be written such that if the image isn't [available](#), and the text is used instead, the text flows seamlessly into the surrounding text, as if the image had never been there in the first place.

#### **4.8.4.4.5 Text that has been rendered to a graphic for typographical effect**

Sometimes, an image just consists of text, and the purpose of the image is not to highlight the actual typographic effects used to render the text, but just to convey the text itself.

In such cases, the [alt](#) attribute must be present but must consist of the same text as written in the image itself.

Considérez un graphique contenant le texte "Jour de la Terre", mais avec les lettres toutes décorées de fleurs et de plantes. Si le texte est simplement utilisé comme titre, pour pimenter la page pour les utilisateurs graphiques, alors le texte alternatif correct est juste le même texte "Jour de la Terre", et aucune mention n'est nécessaire des décorations :

```
<h1></h1>
```

Un manuscrit enluminé peut utiliser des graphiques pour certaines de ses images. Le texte alternatif dans une telle situation est juste le caractère que l'image représente.

```
<p>nce upon a time and a long  
long time ago, late at  
night, when it was dark, over the hills, through the woods, across  
a great ocean, in a land far  
away, in a small house, on a hill, under a full moon...
```

Lorsqu'une image est utilisée pour représenter un caractère qui ne peut pas être représenté autrement en Unicode, par exemple gajji, itaiji, ou de nouveaux caractères tels que de nouveaux symboles monétaires, le texte alternatif doit être une manière plus conventionnelle d'écrire la même chose, par exemple en utilisant le hiragana phonétique ou katakana pour donner la prononciation du caractère.

Dans cet exemple de 1997, un nouveau symbole monétaire qui ressemble à un E bouclé avec deux barres au milieu au lieu d'une est représenté à l'aide d'une image. Le texte alternatif donne la prononciation du caractère.

```
<p>Only 5.99!
```

Une image ne doit pas être utilisée si les caractères servent un objectif identique. Ce n'est que lorsque le texte ne peut pas être directement représenté à l'aide de texte, par exemple, à cause de décorations ou parce qu'il n'y a pas de caractère approprié (comme dans le cas du gajji), qu'une image serait appropriée.

*Si un auteur est tenté d'utiliser une image parce que sa police système par défaut ne prend pas en charge un caractère donné, les polices Web sont une meilleure solution que les images.*

#### **4.8.4.4.6 Une représentation graphique d'une partie du texte environnant**

Dans de nombreux cas, l'image n'est en fait qu'un complément et sa présence ne fait que renforcer le texte qui l'entoure. Dans ces cas, l' alt attribut doit être présent mais sa valeur doit être la chaîne vide.

En général, une image entre dans cette catégorie si la suppression de l'image ne rend pas la page moins utile, mais l'inclusion de l'image permet aux utilisateurs de navigateurs visuels de mieux comprendre le concept.

Un organigramme qui répète le paragraphe précédent sous forme graphique :

```
<p>The Network passes data to the Input Stream Preprocessor, which
passes it to the Tokenizer, which passes it to the Tree
Construction
stage. From there, data goes to both the DOM and to Script
Execution.
Script Execution is linked to the DOM, and, using document.write(),
passes data to the Tokenizer.</p>
<p></p>
```

Dans ces cas, il serait erroné d'inclure un texte alternatif composé uniquement d'une légende. Si une légende doit être incluse, alors soit l' title attribut peut être utilisé, soit les éléments figure et figcaption peuvent être utilisés. Dans ce dernier cas, l'image serait en fait une phrase ou un paragraphe avec une représentation graphique alternative, et nécessiterait donc un texte alternatif.

```
<!-- Using the title="" attribute -->
<p>The Network passes data to the Input Stream Preprocessor, which
passes it to the Tokenizer, which passes it to the Tree
Construction
stage. From there, data goes to both the DOM and to Script
Execution.
Script Execution is linked to the DOM, and, using document.write(),
passes data to the Tokenizer.</p>
<p></p>
<!-- Using <figure> and <figcaption> -->
<p>The Network passes data to the Input Stream Preprocessor, which
passes it to the Tokenizer, which passes it to the Tree
Construction
stage. From there, data goes to both the DOM and to Script
Execution.
Script Execution is linked to the DOM, and, using document.write(),
passes data to the Tokenizer.</p>
<figure>
  
```

```

<figcaption>Flowchart representation of the parsing
model.</figcaption>
</figure>
<!-- This is WRONG. Do not do this. Instead, do what the above
examples do. -->
<p>The Network passes data to the Input Stream Preprocessor, which
passes it to the Tokenizer, which passes it to the Tree
Construction
stage. From there, data goes to both the DOM and to Script
Execution.
Script Execution is linked to the DOM, and, using document.write(),
passes data to the Tokenizer.</p>
<p></p>
<!-- Never put the image's caption in the alt="" attribute! -->

```

Un graphique qui répète le paragraphe précédent sous forme graphique :

```

<p>According to a study covering several billion pages,
about 62% of documents on the web in 2007 triggered the Quirks
rendering mode of web browsers, about 30% triggered the Almost
Standards mode, and about 9% triggered the Standards mode.</p>
<p></p>

```

#### 4.8.4.4.7 Images auxiliaires

Parfois, une image n'est pas essentielle au contenu, mais n'est néanmoins ni purement décorative ni entièrement redondante avec le texte. Dans ces cas, l'alt attribut doit être présent et sa valeur doit être soit la chaîne vide, soit une représentation textuelle des informations que l'image véhicule. Si l'image a une légende donnant le titre de l'image, alors la alt valeur de l'attribut ne doit pas être vide (ce serait assez déroutant pour les lecteurs non visuels).

Prenons l'exemple d'un article de presse sur une personnalité politique, dans lequel le visage de l'individu était représenté sur une image. L'image n'est pas purement décorative, car elle est pertinente pour l'histoire. L'image n'est pas entièrement redondante avec l'histoire non plus, car elle montre à quoi ressemble le politicien. La question de savoir si un texte alternatif doit être fourni est une décision d'auteur, décidée par l'influence de l'image sur l'interprétation de la prose.

Dans cette première variante, l'image est affichée sans contexte, et aucun texte alternatif n'est fourni :

```

<p> Ahead of today's referendum,

```

```
the President wrote an open letter to all registered voters. In it,
she admitted that the country was
divided.</p>
```

Si l'image n'est qu'un visage, il n'y a peut-être aucune valeur à le décrire. Peu importe au lecteur si l'individu a les cheveux roux ou les cheveux blonds, si l'individu a la peau blanche ou la peau noire, si l'individu a un œil ou deux yeux.

Cependant, si l'image est plus dynamique, par exemple montrant le politicien en colère, ou particulièrement heureux, ou dévasté, un texte alternatif serait utile pour donner le ton de l'article, un ton qui pourrait autrement être manqué :

```
<p>
Ahead of today's referendum, the President wrote an open letter to
all
registered voters. In it, she admitted that the country was
divided.
</p>
<p>
Ahead of today's referendum, the President wrote an open letter to
all
registered voters. In it, she admitted that the country was
divided.
</p>
```

Que la personne soit « triste » ou « heureuse » fait une différence dans l'interprétation du reste du paragraphe : dit-elle probablement qu'elle est mécontente de la division du pays, ou dit-elle que la perspective d'un pays divisé est bon pour sa carrière politique ? L'interprétation varie en fonction de l'image.

Si l'image a une légende, l'inclusion d'un texte alternatif évite de laisser l'utilisateur non visuel confus quant à ce à quoi la légende fait référence.

```
<p>Ahead of today's referendum, the President wrote an open letter
to
all registered voters. In it, she admitted that the country was
divided.</p>
<figure>
  
  <figcaption> The President of Ruritania. Photo © 2014 PolitiPhoto.
</figcaption>
</figure>
```

#### **4.8.4.4.8 Une image purement décorative qui n'ajoute aucune information**

Si une image est décorative mais n'est pas particulièrement spécifique à la page - par exemple une image qui fait partie d'un schéma de conception à l'échelle du site - l'image doit être spécifiée dans le CSS du site, pas dans le balisage du document.

Cependant, une image décorative qui n'est pas abordée par le texte environnant mais qui a encore une certaine pertinence peut être incluse dans une page utilisant l' `img` élément. De telles images sont décoratives, mais font toujours partie du contenu. Dans ces cas, l' `alt` attribut doit être présent mais sa valeur doit être la chaîne vide.

Des exemples où l'image est purement décorative bien qu'elle soit pertinente incluraient des choses comme une photo du paysage de Black Rock City dans un article de blog sur un événement à Burning Man, ou une image d'une peinture inspirée d'un poème, sur une page récitant ce poème . L'extrait suivant montre un exemple de ce dernier cas (seul le premier verset est inclus dans cet extrait):

```
<h1>The Lady of Shalott</h1>
<p></p>
<p>On either side the river lie<br>
Long fields of barley and of rye,<br>
That clothe the wold and meet the sky;<br>
And through the field the road run by<br>
To many-tower'd Camelot;<br>
And up and down the people go,<br>
Gazing where the lilies blow<br>
Round an island there below,<br>
The island of Shalott.</p>
```

#### **4.8.4.4.9 Un groupe d'images qui forment une seule image plus grande sans liens**

Lorsqu'une image a été découpée en fichiers image plus petits qui sont ensuite affichés ensemble pour former à nouveau l'image complète, l'une des images doit avoir son `alt` attribut défini selon les règles pertinentes qui seraient appropriées pour l'image dans son ensemble, puis toutes les images restantes doivent avoir leur `alt` attribut défini sur la chaîne vide.

Dans l'exemple suivant, une image représentant un logo d'entreprise pour XYZ Corp a été divisée en deux parties, la première contenant les lettres "XYZ" et la seconde avec le mot "Corp". Le texte alternatif ("XYZ Corp") est tout dans la première image.

```
<h1></h1>
```

Dans l'exemple suivant, une note est affichée sous la forme de trois étoiles pleines et de deux étoiles vides. Alors que le texte alternatif aurait pu être "★★★☆☆", l'auteur a plutôt décidé de donner plus utilement la note sous la forme "3 sur 5". C'est le texte alternatif de la première image, et le reste a un texte alternatif vide.

```
<p>Rating: <meter max=5 value=3></meter></p>
```

#### 4.8.4.4.10 Un groupe d'images qui forment une seule image plus grande avec des liens

En règle générale, [les images cliquables](#) doivent être utilisées au lieu de découper une image pour les liens.

Cependant, si une image est en effet découpée en tranches et que l'un des composants de l'image découpée en tranches est le seul contenu des liens, alors une image par lien doit avoir un texte alternatif dans son `alt` attribut représentant le but du lien.

Dans l'exemple suivant, une image représentant l'emblème du monstre spaghetti volant, avec chacun des appendices noodly gauches et les appendices noodly droits dans différentes images, afin que l'utilisateur puisse choisir le côté gauche ou le côté droit dans une aventure.

```
<h1>The Church</h1>
<p>You come across a flying spaghetti monster. Which side of His
Noodliness do you wish to reach out for?</p>
<p><a href="?go=left" ></a
  ><a href="?go=right"></a></p>
```

#### 4.8.4.4.11 Un élément clé du contenu

Dans certains cas, l'image est une partie essentielle du contenu. Cela peut être le cas, par exemple, sur une page faisant partie d'une galerie de photos. L'image est tout l'*intérêt* de la page qui la contient.

La façon de fournir un texte alternatif pour une image qui est un élément clé du contenu dépend de la provenance de l'image.

## Le cas général

Lorsqu'il est possible de fournir un texte alternatif détaillé, par exemple si l'image fait partie d'une série de captures d'écran dans une critique de magazine, ou fait partie d'une bande dessinée, ou est une photographie dans une entrée de blog à propos de cette photographie, le texte qui peut servir de substitut à l'image doit être donnée comme contenu de l' [alt](#) attribut.

Une capture d'écran dans une galerie de captures d'écran pour un nouveau système d'exploitation, avec un texte alternatif :

```
<figure>
  
  <figcaption>Screenshot of a KDE desktop.</figcaption>
</figure>
```

Un graphique dans un rapport financier :

```

```

Notez que « graphique des ventes » serait un texte de remplacement inadéquat pour un graphique des ventes. Un texte qui serait une bonne *légende* ne convient généralement pas comme texte de remplacement.

## Des images qui défient une description complète

Dans certains cas, la nature de l'image peut être telle qu'il n'est pas pratique de fournir un texte alternatif complet. Par exemple, l'image pourrait être indistincte, ou pourrait être une fractale complexe, ou pourrait être une carte topographique détaillée.



Dans ces cas, l' `alt` attribut doit contenir un texte de remplacement approprié, mais il peut être quelque peu bref.

Parfois, il n'y a tout simplement pas de texte qui puisse rendre justice à une image. Par exemple, on ne peut pas dire grand-chose pour décrire utilement un test de tache d'encre de Rorschach. Cependant, une description, même succincte, vaut toujours mieux que rien :

```
<figure>
  
  <figcaption>A black outline of the first of the ten cards
  in the Rorschach inkblot test.</figcaption>
</figure>
```

Notez que ce qui suit serait une très mauvaise utilisation du texte alternatif :

```
<!-- This example is wrong. Do not copy it. -->
<figure>
  
  <figcaption>A black outline of the first of the ten cards
  in the Rorschach inkblot test.</figcaption>
</figure>
```

Inclure la légende dans le texte alternatif comme celui-ci n'est pas utile car cela duplique efficacement la légende pour les utilisateurs qui n'ont pas d'images, les narguant deux fois sans les aider plus que s'ils n'avaient lu ou entendu la légende qu'une seule fois.

Un autre exemple d'image qui défie toute description est une fractale, qui, par définition, est infinie dans les détails.

L'exemple suivant montre une manière possible de fournir un texte alternatif pour la vue complète d'une image de l'ensemble de Mandelbrot.

```


De même, une photographie du visage d'une personne, par exemple dans une biographie, peut être considérée comme assez pertinente et essentielle pour le contenu, mais il peut être difficile de substituer entièrement du texte à :

```
<section class="bio">
  <h1>A Biography of Isaac Asimov</h1>
  <p>Born <b>Isaak Yudovich Ozimov</b> in 1920, Isaac was a prolific author.</p>
  <p></p>
  <p>Asimov was born in Russia, and moved to the US when he was three years old.</p>
  <p>...</p>
</section>
```

Dans de tels cas, il est inutile (et même déconseillé) d'inclure une référence à la présence de l'image elle-même dans le texte alternatif, car ce texte serait redondant, le navigateur lui-même signalant la présence de l'image. Par exemple, si le texte alternatif était "Une photo d'Isaac Asimov", alors un agent utilisateur conforme pourrait le lire comme "(Image) Une photo d'Isaac Asimov" plutôt que le plus utile "(Image) Isaac Asimov avait les cheveux noirs, un front haut et portait des lunettes...".

## Images dont le contenu n'est pas connu

Dans certains cas malheureux, il peut n'y avoir aucun texte alternatif disponible, soit parce que l'image est obtenue de manière automatisée sans aucun texte alternatif associé (par exemple, une webcam), soit parce que la page est générée par un script utilisant user- fourni des images où l'utilisateur n'a pas fourni de texte alternatif approprié ou utilisable (par exemple, des sites de partage de photographies), ou parce que l'auteur ne sait pas lui-même ce que les images représentent (par exemple, un photographe aveugle partageant une image sur son blog).

Dans de tels cas, l' `alt` attribut peut être omis, mais l'une des conditions suivantes doit également être remplie :

- L' `img` élément se trouve dans un `figure` élément qui contient un `figcaption` élément qui contient un contenu autre que [l'espace blanc inter-élément](#) et, en ignorant l' `figcaption` élément et ses descendants, l' `figure` élément n'a pas de descendants [de contenu de flux](#) autres que [l'espace blanc inter-élément](#) et l' `img` élément.

- L' title attribut est présent et a une valeur non vide.

*Il est actuellement déconseillé de s'appuyer sur l' title attribut car de nombreux agents utilisateurs n'exposent pas l'attribut de manière accessible comme l'exige cette spécification (par exemple, en exigeant un dispositif de pointage tel qu'une souris pour faire apparaître une info-bulle, ce qui exclut les utilisateurs utilisant uniquement le clavier et les touches tactiles). -uniquement les utilisateurs, comme toute personne disposant d'un téléphone ou d'une tablette moderne).*

*Ces cas doivent être réduits au strict minimum. S'il y a la moindre possibilité que l'auteur ait la capacité de fournir un véritable texte alternatif, alors il ne serait pas acceptable d'omettre l' alt attribut.*

Une photo sur un site de partage de photos, si le site a reçu l'image sans métadonnées autres que la légende, pourrait être balisée comme suit :

```
<figure>
  
  <figcaption>Bubbles traveled everywhere with
us.</figcaption>
</figure>
```

Il serait préférable, cependant, qu'une description détaillée des parties importantes de l'image soit obtenue de l'utilisateur et incluse sur la page.

Le blog d'un utilisateur aveugle dans lequel une photo prise par l'utilisateur est affichée. Au départ, l'utilisateur peut ne pas avoir la moindre idée de ce que montre la photo qu'il a prise :

```
<article>
  <h1>I took a photo</h1>
  <p>I went out today and took a photo!</p>
  <figure>
    
    <figcaption>A photograph taken blindly from my front
porch.</figcaption>
  </figure>
</article>
```

Finalement, cependant, l'utilisateur pourrait obtenir une description de l'image de ses amis et pourrait alors inclure un texte alternatif :

```
<article>
  <h1>I took a photo</h1>
  <p>I went out today and took a photo!</p>
  <figure>
    
    <figcaption>A photograph taken blindly from my front
    porch.</figcaption>
  </figure>
</article>

```

Parfois, tout l'intérêt de l'image est qu'une description textuelle n'est pas disponible, et l'utilisateur doit fournir la description. Par exemple, le but d'une image CAPTCHA est de voir si l'utilisateur peut littéralement lire le graphique. Voici une façon de baliser un CAPTCHA (notez l' titleattribut):

```

<p><label>What does this image say?

<input type="text" name="captcha"></label>
(If you cannot see the image, you can use an <a
href="?audio">audio</a> test instead.)</p>

```

Un autre exemple serait un logiciel qui affiche des images et demande un texte alternatif précisément dans le but d'écrire ensuite une page avec un texte alternatif correct. Une telle page pourrait avoir un tableau d'images, comme celui-ci :

```

<table>
  <thead>
    <tr> <th> Image <th> Description
  <tbody>
    <tr>
      <td> 
      <td> <input name="alt2421">
    <tr>
      <td> 
      <td> <input name="alt2422">
    </table>

```

Notez que même dans cet exemple, autant d'informations utiles que possible sont toujours incluses dans l' titleattribut.

*Étant donné que certains utilisateurs ne peuvent pas du tout utiliser les images (par exemple, parce qu'ils ont une connexion très lente, ou parce qu'ils utilisent un navigateur texte uniquement, ou parce qu'ils écoutent la page lue*

*par un navigateur Web vocal automobile mains libres, ou simplement parce qu'ils sont aveugles), l' alt attribut ne peut être omis plutôt que d'être fourni avec un texte de remplacement lorsqu'aucun texte alternatif n'est disponible et qu'aucun ne peut être rendu disponible, comme dans les exemples ci-dessus. Le manque d'effort de la part de l'auteur n'est pas une raison acceptable pour omettre l' alt attribut.*

#### **4.8.4.4.12 Une image non destinée à l'utilisateur**

En règle générale, les auteurs doivent éviter d'utiliser img des éléments à des fins autres que l'affichage d'images.

Si un img élément est utilisé à des fins autres que l'affichage d'une image, par exemple dans le cadre d'un service pour compter les pages vues, l' alt attribut doit être la chaîne vide.

Dans ce cas, les attributs width et height doivent tous deux être définis sur zéro.

#### **4.8.4.4.13 Une image dans un e-mail ou un document privé destiné à une personne spécifique connue pour être en mesure de visualiser des images**

*Cette section ne s'applique pas aux documents accessibles au public ou dont le public cible n'est pas nécessairement personnellement connu de l'auteur, tels que les documents sur un site Web, les e-mails envoyés à des listes de diffusion publiques ou la documentation du logiciel.*

Lorsqu'une image est incluse dans une communication privée (telle qu'un e-mail HTML) destinée à une personne spécifique connue pour être en mesure de visualiser les images, l' alt attribut peut être omis. Cependant, même dans de tels cas, les auteurs sont fortement invités à inclure un texte alternatif (selon le type d'image impliqué, comme décrit dans les entrées ci-dessus), afin que l'e-mail soit toujours utilisable si l'utilisateur utilise un client de messagerie qui ne prend en charge les images, ou le document doit-il être transmis à d'autres utilisateurs dont les capacités pourraient ne pas inclure la visualisation facile des images.

#### **4.8.4.4.14 Conseils pour les générateurs de balisage**

Les générateurs de balisage (tels que les outils de création WYSIWYG) doivent, dans la mesure du possible, obtenir un texte alternatif de leurs

utilisateurs. Cependant, il est reconnu que dans de nombreux cas, cela ne sera pas possible.

Pour les images qui sont le seul contenu des liens, les générateurs de balisage doivent examiner la cible du lien pour déterminer le titre de la cible, ou l'URL de la cible, et utiliser les informations obtenues de cette manière comme texte alternatif.

Pour les images qui ont des légendes, les générateurs de balisage doivent utiliser les éléments `figure` et `figcaption`, ou l'attribut `title`, pour fournir la légende de l'image.

En dernier recours, les implémenteurs doivent soit définir l'attribut `alt` sur la chaîne vide, en supposant que l'image est une image purement décorative qui n'ajoute aucune information mais reste spécifique au contenu environnant, soit omettre complètement l'attribut, sous l'hypothèse que l'image est un élément clé du contenu.

Les générateurs de balisage peuvent spécifier un attribut `generator-unable-to-provide-required-alt` sur les éléments pour lesquels ils n'ont pas pu obtenir de texte alternatif et pour lesquels ils ont donc omis l'attribut `alt`. La valeur de cet attribut doit être la chaîne vide. Les documents contenant de tels attributs ne sont pas conformes, mais les vérificateurs de conformité ignoreront silencieusement cette erreur.

*Ceci est destiné à éviter que les générateurs de balisage ne soient contraints de remplacer l'erreur consistant à omettre l'attribut `alt` par l'erreur encore plus flagrante consistant à fournir un texte alternatif bidon, car les vérificateurs de conformité automatisés de pointe ne peuvent pas distinguer le texte alternatif bidon du texte alternatif correct. .*

Les générateurs de balisage doivent généralement éviter d'utiliser le nom de fichier de l'image comme texte alternatif. De même, les générateurs de balisage devraient éviter de générer du texte alternatif à partir de tout contenu qui sera également disponible pour les agents utilisateurs de présentation (par exemple, les navigateurs Web).

*En effet, une fois qu'une page est générée, elle ne sera généralement pas mise à jour, alors que les navigateurs qui lisent la page ultérieurement peuvent être mis à jour par l'utilisateur. Par conséquent, le navigateur est susceptible d'avoir des heuristiques plus à jour et plus précises que le générateur de balisage a fait lors de la génération de la page.*

#### **4.8.4.4.15 Conseils pour les vérificateurs de conformité**

Un vérificateur de conformité doit signaler l'absence d'un attribut `alt` comme une erreur, sauf si l'une des conditions répertoriées ci-dessous s'applique :

- L' [img](#) élément se trouve dans un [figure](#) élément qui satisfait [aux conditions décrites ci-dessus](#) .
- L' [img](#) élément a un [title](#) attribut avec une valeur qui n'est pas la chaîne vide (également comme [décrit ci-dessus](#) ).
- Le vérificateur de conformité a été configuré pour supposer que le document est un e-mail ou un document destiné à une personne spécifique connue pour pouvoir afficher des images.
- L' [img](#) élément a un attribut (non conforme) [generator-unable-to-provide-required-alt](#) dont la valeur est la chaîne vide. Un vérificateur de conformité qui ne signale pas l'absence d'un [alt](#) attribut comme une erreur ne doit pas non plus signaler la présence de l' [generator-unable-to-provide-required-alt](#) attribut vide comme une erreur. (Ce cas ne représente pas un cas où le document est conforme, seulement que le générateur n'a pas pu déterminer le texte alternatif approprié - les validateurs ne sont pas tenus de montrer une erreur dans ce cas, car une telle erreur pourrait encourager les générateurs de balisage à inclure un faux texte alternatif dans le seul but de faire taire les validateurs. Naturellement, les vérificateurs de conformité *peuvent* signaler l'absence d'un [alt](#) attribut comme une erreur, même en présence du [generator-unable-to-provide-required-alt](#) attribut; par exemple, il pourrait y avoir une option utilisateur pour signaler *toutes* les erreurs de conformité, même celles qui pourraient être le résultat plus ou moins inévitable de l'utilisation d'un générateur de balisage.)

#### 4.8.5 L' [iframe](#) élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu intégré](#) .  
[Contenu interactif](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

##### Modèle de contenu :

[Rien](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

### Attributs globaux

src— Adresse de la ressource

srcdoc— Un document à rendre dans le iframe

name— Nom du contenu navigable

sandbox— Règles de sécurité pour le contenu imbriqué

allow— Politique d'autorisations à appliquer au iframe contenu de .

allowfullscreen— S'il faut autoriser iframe l'utilisation du contenu de requestFullscreen()

width— Dimensions horizontales

height— Dimension verticale

referrerpolicy— Politique de référence pour les récupérations initiées par l'élément

loading— Utilisé lors de la détermination du report de chargement

## Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

## Interface DOM :

[Exposed=Window]

interface **HTMLIFrameElement** : [HTMLElement](#) {

[[HTMLConstructor](#)] constructor();

[[CEReactions](#)] attribute USVString [src](#);

[[CEReactions](#)] attribute DOMString [srcdoc](#);

[[CEReactions](#)] attribute DOMString [name](#);

[SameObject, PutForwards=[value](#)] readonly attribute

[DOMTokenList](#) [sandbox](#);

[[CEReactions](#)] attribute DOMString [allow](#);

[[CEReactions](#)] attribute boolean [allowFullscreen](#);

[[CEReactions](#)] attribute DOMString [width](#);

[[CEReactions](#)] attribute DOMString [height](#);

[[CEReactions](#)] attribute DOMString [referrerPolicy](#);



```

[CEReactions] attribute DOMString loading;

readonly attribute Document? contentDocument;

readonly attribute WindowProxy? contentWindow;

Document? getSVGDocument() ;

// also has obsolete members

};

```

L' [iframe](#) élément [représente](#) son [contenu navigable](#) .

L' [src](#) attribut donne l' [URL](#) d'une page que le [contenu navigable](#) de l'élément doit contenir. L'attribut, s'il est présent, doit être une [URL non vide valide potentiellement entourée d'espaces](#) . Si l' [itemprop](#) attribut est spécifié sur un [iframe](#) élément, alors l' [src](#) attribut doit également être spécifié.

✓ MDN

L' [srcdoc](#) attribut donne le contenu de la page que le [contenu navigable](#) de l'élément doit contenir. La valeur de l'attribut est la source d' **un [iframe srcdoc](#) document** .

L' [srcdoc](#) attribut, s'il est présent, doit avoir une valeur utilisant [la syntaxe HTML](#) composée des composants syntaxiques suivants, dans l'ordre indiqué :

1. N'importe quel nombre de [commentaires](#) et [d'espaces ASCII](#) .
2. Facultativement, un [DOCTYPE](#) .
3. N'importe quel nombre de [commentaires](#) et [d'espaces ASCII](#) .
4. L' [élément document](#) , sous la forme d'un [html élément](#) .
5. N'importe quel nombre de [commentaires](#) et [d'espaces ASCII](#) .

*Les exigences ci-dessus s'appliquent également aux [documents XML](#) .*

Ici, un blog utilise l' [srcdoc](#) attribut en conjonction avec l' [sandbox](#) attribut décrit ci-dessous pour fournir aux utilisateurs d'agents utilisateurs qui prennent en charge cette fonctionnalité une couche supplémentaire de protection contre l'injection de script dans les commentaires des articles de blog :

```

<article>
  <h1>I got my own magazine!</h1>
  <p>After much effort, I've finally found a publisher, and so now I

```

```

have my own magazine! Isn't that awesome?! The first issue will
come

out in September, and we have articles about getting food, and
about

getting in boxes, it's going to be great!</p>
<footer>

  <p>Written by <a href="/users/cap">cap</a>, 1 hour ago.
</footer>

<article>

  <footer> Thirteen minutes ago, <a href="/users/ch">ch</a> wrote:
</footer>

  <iframe sandbox srcdoc="<p>did you get a cover picture
yet?"></iframe>

</article>

<article>

  <footer> Nine minutes ago, <a href="/users/cap">cap</a> wrote:
</footer>

  <iframe sandbox srcdoc="<p>Yeah, you can see it <a
href=&quot;/gallery?mode=cover&amp;amp;page=1&quot;>in my
gallery</a>."></iframe>

</article>

<article>

  <footer> Five minutes ago, <a href="/users/ch">ch</a> wrote:
</footer>

  <iframe sandbox srcdoc="<p>hey that's earl's table.
<p>you should get earl&amp;amp;me on the next cover."></iframe>

</article>

```

Notez la façon dont les guillemets doivent être échappés (sinon l' `srcdoc` attribut se terminerait prématurément), et la façon dont les esperluettes brutes (par exemple dans les URL ou en prose) mentionnées dans le contenu du bac à sable doivent être échappées *deux fois* - une fois pour que l'esperluette soit préservée lorsque à l'origine de l'analyse de l' `srcdoc` attribut, et une fois de plus pour éviter que l'esperluette ne soit mal interprétée lors de l'analyse du contenu du bac à sable.

De plus, notez que puisque le `DOCTYPE` est facultatif dans `iframe srcdoc documents`, et que les éléments `html`, `head` et `body` ont des balises de début et de fin facultatives, et que l' `title` élément est également facultatif dans `iframe srcdoc documents`, le balisage d'un `srcdoc` attribut peut être relativement succinct malgré la représentation d'un document entier, puisque seul le contenu de l' `body` élément doit apparaître littéralement dans la syntaxe. Les autres éléments sont toujours présents, mais seulement par implication.

*Dans la syntaxe HTML, les auteurs n'ont qu'à se rappeler d'utiliser les caractères U+0022 Guillemets (") pour envelopper le contenu de l'attribut, puis d'échapper tous les caractères U+0026 AMPERSAND (&) et U+0022 Guillemets ("), et de spécifier l' `sandbox` attribut, pour garantir une intégration sécurisée du contenu. (Et n'oubliez*

*pas d'échapper les esperluettes avant les guillemets, pour vous assurer que les guillemets deviennent " et non &quot;.)*  
*En XML, le caractère U+003C LESS-THAN SIGN (<) doit également être échappé. Afin d'empêcher [la normalisation des valeurs d'attribut](#), certains des caractères d'espacement XML - en particulier U+0009 CHARACTER TABULATION (tabulation), U+000A LINE FEED (LF) et U+000D CARRIAGE RETURN (CR) - doivent également être échappés. [\[XML\]](#)*  
*Si l'[src](#)attribut et l'[srcdoc](#)attribut sont spécifiés ensemble, l'[srcdoc](#)attribut est prioritaire. Cela permet aux auteurs de fournir une [URL](#) de secours pour les agents utilisateurs hérités qui ne prennent pas en charge l'[srcdoc](#)attribut.*

---

Lorsqu'un [iframe](#)élément est inséré [dans un document](#) dont [le contexte de navigation](#) est non nul, l'agent utilisateur doit exécuter ces étapes :

1. [Créez un nouvel enfant navigable](#) pour l'élément .
2. Si l'élément a un [sandbox](#)attribut, [analysez la directive de sandboxing](#) en fonction de la valeur de l'attribut et de l'indicateur de sandboxing de l'élément [iframe](#)[set](#) .
3. [Traitez les \[iframe\]\(#\)attributs](#) de *element* , avec [initialInsertion](#) défini sur true.

Lorsqu'un [iframe](#)élément est [supprimé d'un document](#) , l'agent utilisateur doit [détruire un enfant navigable](#) compte tenu de l'élément.

*Cela se produit sans qu'aucun [unload](#)événement ne se déclenche (le [contenu de l'élément est navigable](#) et le sien [Document](#)est [détruit](#) , pas [déchargé](#) ).*

Chaque fois qu'un élément avec un [contenu navigable](#)[iframe](#) non nul voit son attribut défini, modifié ou supprimé, l'agent utilisateur doit [traiter les attributs](#) [.srcdociframe](#)

De même, chaque fois qu'un élément avec un [contenu](#)[iframe](#) non nul navigable mais sans attribut spécifié voit son attribut défini, modifié ou supprimé, l'agent utilisateur doit [traiter les attributs](#) [.srcdocsrciframe](#)

Pour **traiter les [iframe](#)attributs** d'un élément element , avec un booléen optionnel ***initialInsertion*** (false par défaut) :

1. Si l'attribut de l' *élément*[srcdoc](#) est spécifié, alors :
  1. La navigation actuelle de l'élément défini [était](#) un booléen chargé paresseusement sur faux.

2. Si les étapes de chargement paresseux de l'élément donné renvoient la valeur true, alors :
  1. Définissez les étapes de reprise de chargement paresseux de l'élément sur le reste de cet algorithme en commençant par l'étape intitulée *naviguer vers la ressource srcdoc*.
  2. La navigation actuelle de l'élément Set était un booléen chargé paresseux sur true.
  3. Commencez à observer l'intersection d'un élément de chargement paresseux pour l'élément.
  4. Retour.
3. Naviguez jusqu'à la ressource srcdoc : parcourez un iframe ou un frame élément donné, about:srcdoc, la chaîne vide et la valeur de l'attribut de l'élément srcdoc.

Le résultat Document doit être considéré comme un iframe srcdoc document.

2. Sinon:

1. Soit *url* le résultat de l'exécution des étapes de traitement des attributs partagés pour iframe et frame les éléments donnés *element* et *initialInsertion*.
2. Si l'URL est nulle, alors retournez.
3. Si *url* correspond about:blank et *initialInsertion* est vrai, alors :
  1. Exécutez les étapes de l'événement de chargement d'iframe donné *element*.
  2. Retour.
4. Soit *referrerPolicy* l'état actuel de l'attribut content de l'élément referrerpolicy.
5. La navigation actuelle de l'élément défini était un booléen chargé paresseusement sur faux.
6. Si les étapes de chargement paresseux de l'élément donné renvoient la valeur true, alors :
  1. Définissez les étapes de reprise de chargement paresseux de l'élément sur le reste de cet algorithme en commençant par l'étape étiquetée *naviguer*.

2. La navigation actuelle de l'élément Set était un booléen chargé paresseux sur true.
3. Commencez à observer l'intersection d'un élément de chargement paresseux pour l'élément .
4. Retour.
7. Naviguer : naviguer dans un `iframe` élément `frame` donné , une URL et `referrerPolicy` .

Les étapes de traitement des attributs partagés pour les éléments `iframe` et `frame` , étant donné un élément `element` et un booléen `initialInsertion` , sont :

1. Soit `url` l' enregistrement d'URL `about:blank` .
2. Si `element` a un `src` attribut spécifié et que sa valeur n'est pas la chaîne vide, alors analysez la valeur de cet attribut par rapport au nœud document de l' élément . Si cela réussit, définissez `url` sur l' enregistrement d'URL résultant .
3. Si l' ancêtre inclusif navigables du nœud navigable de l'élément contient un navigable dont l'URL du document actif est égale à `url` avec les fragments exclus définis sur true, alors renvoie null.
4. Si `url` correspond `about:blank` et `initialInsertion` est vrai, alors effectuez les étapes de mise à jour de l'URL et de l'historique en fonction du document actif et de l'`url` de l' élément navigable de contenu .

*Ceci est nécessaire dans le cas où l'url est quelque chose comme `about:blank?foo`. Si url est tout simplement `about:blank`, cela ne fera rien.*

5. URL de retour .

Pour **naviguer dans un `iframe` ou `frame`** donné un élément `element` , une URL `url` , une politique de référence `referrerPolicy` et une chaîne ou une valeur facultative `srcdocString` (nulle par défaut) :

1. Soit `historyHandling` la valeur " push ".
2. Si le document actif du contenu navigable de l' élément n'est pas complètement chargé , définissez `historyHandling` sur " " . replace
3. Naviguez dans le contenu de l' élément navigable vers `url` en utilisant le document de nœud de l'élément , avec historyHandling défini sur `historyHandling` , referrerPolicy défini sur `referrerPolicy` et documentResource défini sur `srcdocString` .

Chacun Document a un indicateur **de chargement iframe en cours** et un indicateur **de chargement iframe muet** . Quand a Document est créé, ces drapeaux doivent être désactivés pour cela Document.

Pour exécuter les **étapes de l'événement iframe load** , étant donné un élément iframe element :

1. Assert : le contenu navigable de l'élément n'est pas nul.
2. Soit *childDocument* le document actif du contenu navigable de l'élément .
3. Si *childDocument* a son indicateur de chargement d'iframe muet défini, alors retournez.
4. Définissez l'indicateur de chargement iframe en cours de *childDocument* .
5. Lancez un événement nommé load à l'élément .
6. Désactivez l'indicateur de chargement iframe en cours de *childDocument* .

***Ceci, en conjonction avec les scripts, peut être utilisé pour sonder l'espace URL des serveurs HTTP du réseau local. Les agents utilisateurs peuvent implémenter des politiques de contrôle d'accès cross-origin qui sont plus strictes que celles décrites ci-dessus pour atténuer cette attaque, mais malheureusement, ces politiques ne sont généralement pas compatibles avec le contenu Web existant.***

Si un type d'élément **retarde potentiellement l'événement load** , alors pour chaque élément element de ce type, l'agent utilisateur doit retarder l'événement load du document de nœud de l'élément si le contenu navigable de l'élément n'est pas nul et que l'un des éléments suivants est vrai :

- Le document actif du contenu navigable de l'élément n'est pas prêt pour les tâches de post-chargement .
- le contenu navigable de l'élément retarde load les événements est vrai.
- Tout retarde l'événement de chargement du document actif du contenu navigable de l'élément .

*Si, lors de la gestion de l' load événement, le contenu navigable de l'élément est à nouveau parcoursu , cela retardera encore plus l'événement load .*

Chaque iframe élément a une **navigation actuelle associée** a été booléen chargé  **paresseux** , initialement faux. Il est activé et désactivé dans le processus de l'algorithme iframe des attributs .

Un iframe élément dont la navigation actuelle était chargée paresseusement booléen est faux retarde potentiellement l'événement load .

*Si, lors de la création de l'élément, l' srcdoc attribut n'est pas défini et que l' src attribut n'est pas non plus défini ou défini mais que sa valeur ne peut pas être analysée , le contenu navigable de l'élément restera à l' initiale about:blank Document .*  
*Si l'utilisateur quitte cette page, l' objet actif iframe de ' contenu navigable ' remplacera les nouveaux objets par de nouveaux objets, mais l' attribut ne changera pas.*WindowProxyWindowDocumentsrc

---

L' **name** attribut, s'il est présent, doit être un nom de cible navigable valide . La valeur donnée est utilisée pour nommer le contenu de l'élément navigable s'il est présent lors de sa création .

---



L' **sandbox** attribut, lorsqu'il est spécifié, active un ensemble de restrictions supplémentaires sur tout contenu hébergé par le iframe. Sa valeur doit être un ensemble non ordonné de jetons uniques séparés par des espaces qui ne respectent pas la casse ASCII . Les valeurs autorisées sont :

- allow-downloads
- allow-forms
- allow-modals
- allow-orientation-lock
- allow-pointer-lock
- allow-popups
- allow-popups-to-escape-sandbox
- allow-presentation
- allow-same-origin
- allow-scripts
- allow-top-navigation
- allow-top-navigation-by-user-activation
- allow-top-navigation-to-custom-protocols

Lorsque l'attribut est défini, le contenu est traité comme provenant d'une origine unique , les formulaires, les scripts et diverses API potentiellement gênantes sont désactivés et les liens ne peuvent pas cibler d'autres éléments navigables . Le allow-same-origin mot-clé fait que le contenu est traité comme



provenant de sa véritable origine au lieu de le forcer à une origine unique ; le `allow-top-navigation` mot clé permet au contenu de [naviguer](#) dans son [navigable traversable](#) ; le `allow-top-navigation-by-user-activation` mot-clé se comporte de manière similaire mais n'autorise une telle [navigation](#) que lorsque la [fenêtre active](#) du contexte de navigation a une [activation transitoire](#) ; le `allow-top-navigation-to-custom-protocols` permet de réactiver les navigations vers [le schéma de non-récupération](#) à [transmettre à un logiciel externe](#) ; et les mots-clés `allow-forms`, `allow-modals`, `allow-orientation-lock`, `allow-pointer-lock`, `allow-popups`, `allow-presentation`, `allow-scripts` et `allow-popups-to-escape-sandbox` réactivent respectivement les formulaires, les boîtes de dialogue modales, le verrouillage de l'orientation de l'écran, l'API de verrouillage du pointeur, les fenêtres contextuelles, l'API de présentation, les scripts et la création de contextes de navigation auxiliaires sans bac à [sable](#) . Le `allow-downloads` mot clé permet au contenu d'effectuer des téléchargements. [\[POINTERLOCK\]](#) [\[SCREENORIENTATION\]](#) [\[PRÉSENTATION\]](#)

Les mots-clés `allow-top-navigation` et `allow-top-navigation-by-user-activation` ne doivent pas être spécifiés tous les deux, car cela est redondant ; n'aura `allow-top-navigation` d'effet que sur un tel balisage non conforme.

De même, le `allow-top-navigation-to-custom-protocols` mot-clé ne doit pas être spécifié si `allow-top-navigation` ou `allow-popups` est spécifié, car cela est redondant.

*Pour autoriser `alert()`, `confirm()` et `prompt()` à l'intérieur du contenu en bac à sable, les mots-clés `allow-modals` et `allow-same-origin` doivent être spécifiés, et l'URL chargée doit avoir [la même origine](#) que l'[origine de niveau supérieur](#) . Sans le `allow-same-origin` mot-clé, le contenu est toujours traité comme d'origine croisée et le contenu d'origine croisée [ne peut pas afficher de boîtes de dialogue simples](#) .*

**Définir à la fois les mots-clés `allow-scripts` et `allow-same-origin` lorsque la page intégrée a la [même origine](#) que la page contenant le `iframe` permet à la page intégrée de supprimer simplement l' `sandbox` attribut, puis de se recharger, sortant ainsi complètement du bac à sable.**

**Ces drapeaux ne prennent effet que lorsque le [contenu navigable](#) de l'`iframe` élément est [navigué](#) . Les supprimer, ou supprimer l'intégralité `sandbox` de l'attribut, n'a aucun effet sur une page déjà chargée. Les fichiers potentiellement hostiles ne doivent pas être servis depuis le même serveur que le fichier contenant l' `iframe` élément. Le contenu hostile de `sandbox` est d'une aide minimale si un attaquant peut convaincre l'utilisateur de simplement visiter le contenu hostile directement, plutôt que dans le fichier `iframe` . Pour limiter les dommages pouvant être causés par un contenu HTML hostile, il doit être diffusé à partir d'un domaine dédié distinct. L'utilisation d'un domaine différent garantit que les scripts dans les fichiers ne peuvent pas attaquer le site, même si l'utilisateur est amené à visiter ces pages directement, sans la protection de l' `sandbox` attribut.**



Lorsque l'attribut `iframe` d'un élément est défini ou modifié alors qu'il a un contenu `sandbox` non nul navigable, l'agent utilisateur doit [analyser la directive de sandboxing](#) en fonction de la valeur de l'attribut et de l' [indicateur de sandboxing](#) de l'élément set `.iframeiframe`

Lorsque l'attribut `iframe` d'un élément `sandbox` est supprimé alors qu'il a un contenu non nul [navigable](#), l'agent utilisateur doit vider l' [ensemble d'indicateurs de sandboxing](#) `iframe` de l'élément `.iframe`

Dans cet exemple, du contenu HTML complètement inconnu, potentiellement hostile, fourni par l'utilisateur est intégré dans une page. Parce qu'il est servi à partir d'un domaine distinct, il est affecté par toutes les restrictions intersites normales. De plus, la page intégrée a des scripts désactivés, des plug-ins désactivés, des formulaires désactivés, et elle ne peut naviguer dans aucun cadre ou fenêtre autre qu'elle-même (ou tout cadre ou fenêtre qu'elle intègre elle-même).

```
<p>We're not scared of you! Here is your content, unedited:</p>
<iframe sandbox
src="https://usercontent.example.net/getusercontent.cgi?id=12193"><
/iframe>
```

***Il est important d'utiliser un domaine distinct afin que si l'attaquant convainc l'utilisateur de visiter cette page directement, la page ne s'exécute pas dans le contexte d'origine du site, ce qui rendrait l'utilisateur vulnérable à toute attaque trouvée dans la page.***

Dans cet exemple, un gadget d'un autre site est intégré. Le gadget a des scripts et des formulaires activés, et les restrictions du bac à sable d'origine sont levées, ce qui permet au gadget de communiquer avec son serveur d'origine. Cependant, le bac à sable est toujours utile, car il désactive les plug-ins et les fenêtres contextuelles, réduisant ainsi le risque que l'utilisateur soit exposé à des logiciels malveillants et à d'autres désagréments.

```
<iframe sandbox="allow-same-origin allow-forms allow-scripts"
src="https://maps.example.com/embedded.html"></iframe>
```

Supposons qu'un fichier A contienne le fragment suivant :

```
<iframe sandbox="allow-same-origin allow-forms" src=B></iframe>
```

Supposons que le fichier B contienne également une iframe :

```
<iframe sandbox="allow-scripts" src=C></iframe>
```

De plus, supposons que le fichier C contienne un lien :

```
<a href=D>Link</a>
```

Pour cet exemple, supposons que tous les fichiers ont été servis en tant que [text/html](#).

La page C dans ce scénario a tous les indicateurs de sandboxing définis. Les scripts sont désactivés, car le `iframe` en A a des scripts désactivés, ce qui remplace le `allow-scripts` mot-clé défini sur le `iframe` en B. Les formulaires sont également désactivés, car le mot-clé `iframe` n'est pas `allow-forms` défini pour l'interne (in B).

Supposons maintenant qu'un script dans A supprime tous les `sandbox` attributs dans A et B. Cela ne changerait rien immédiatement. Si l'utilisateur cliquait sur le lien dans C, chargeant la page D dans dans `iframe` B, la page D agirait désormais comme si dans `iframe` B avait les mots-clés `allow-same-origin` et `allow-forms` définis, car c'était l'état du `contenu navigable` dans dans `iframe` A lorsque la page B était chargé.

De manière générale, supprimer ou modifier dynamiquement l' `sandbox` attribut est déconseillé, car il peut être assez difficile de raisonner sur ce qui sera autorisé et ce qui ne le sera pas.

---

L' `allow` attribut, lorsqu'il est spécifié, détermine la [stratégie de conteneur](#) qui sera utilisée lorsque la [stratégie d'autorisations](#) pour a `Document` dans le `iframe` contenu [navigable](#) de est initialisée. Sa valeur doit être une [stratégie d'autorisations sérialisée](#) . [\[POLITIQUE D'AUTORISATION\]](#)

Dans cet exemple, un `iframe` est utilisé pour intégrer une carte à partir d'un service de navigation en ligne. L' `allow` attribut est utilisé pour activer l'API de géolocalisation dans le contexte imbriqué.

```
<iframe src="https://maps.example.com/"
allow="geolocation"></iframe>
```

L' `allowfullscreen` attribut est un [attribut booléen](#) . Lorsqu'il est spécifié, il indique que `Document` les objets du `contenu navigable` `iframe` de l'élément seront initialisés avec une [politique d'autorisations](#) permettant à la " " fonctionnalité d'être utilisée depuis n'importe quelle [origine](#) . Ceci est appliqué par l' algorithme [des attributs de politique d'autorisations de processus](#) . [\[POLITIQUE D'AUTORISATION\]](#)`fullscreen`

Ici, un `iframe` est utilisé pour intégrer un lecteur à partir d'un site vidéo. L' `allowfullscreen` attribut est nécessaire pour permettre au lecteur d'afficher sa vidéo en plein écran.

```
<article>
  <header>
    <p> <b>Fred
    Flintstone</b></p>
```

```

<p><a href="/posts/3095182851" rel=bookmark>12:44</a> – <a
href="#acl-3095182851">Private Post</a></p>

</header>

<p>Check out my new ride!</p>

<iframe src="https://video.example.com/embed?id=92469812"
allowfullscreen></iframe>

</article>

```

*Ni allow ni ne allowfullscreen peut accorder l'accès à une fonctionnalité dans le contenu navigable iframe d'un élément si le document de nœud de l'élément n'est pas déjà autorisé à utiliser cette fonctionnalité.*

Pour déterminer si un document d'objet est **autorisé à utiliser** la fonction de fonctionnalité contrôlée par des règles, exécutez ces étapes :

1. Si le contexte de navigation du document est nul, renvoie false.
2. Si le document n'est pas complètement actif, alors retourne false.
3. Si le résultat de l'exécution est la fonctionnalité activée dans le document pour l'origine sur la fonctionnalité, le document et l'origine du document est " ", alors renvoie true.  
Enabled
4. Renvoie faux.

*Parce qu'ils n'influencent que la politique d'autorisations du document actif du contenu navigable, les attributs et ne prennent effet que lorsque le contenu navigable du est navigué. Leur ajout ou leur suppression n'a aucun effet sur un document déjà chargé.*  
allowallowfullscreeniframe

---

L' iframe élément prend en charge les attributs de dimension pour les cas où le contenu intégré a des dimensions spécifiques (par exemple, les blocs d'annonces ont des dimensions bien définies).

Un iframe élément n'a jamais de contenu de secours, car il créera toujours un nouvel enfant navigable, que le contenu initial spécifié soit utilisé avec succès ou non.

---

L' **referrerpolicy** attribut est un [attribut de stratégie de référence](#) . Son but est de définir la [politique de référence](#) utilisée lors [du traitement des `iframe` attributs](#) . [\[POLITIQUE DE RÉFÉRENCE\]](#)

L' **loading** attribut est un [attribut de chargement différé](#) . Son but est d'indiquer la politique de chargement [iframe](#) des éléments qui se trouvent en dehors de la fenêtre.

Lorsque l' [loading](#) état de l'attribut passe à l' état [Impatient](#) , l'agent utilisateur doit exécuter ces étapes :

1. Soit *resumptionSteps* les [étapes de reprise de chargement paresseux](#)[iframe](#) de l'élément .
2. Si *ResumptionSteps* est null, alors retournez.
3. Définissez les [étapes de reprise de chargement paresseux](#)[iframe](#) de sur null.
4. Appelez *resumptionSteps* .

---

Les descendants des [iframe](#) éléments ne représentent rien. (Dans les anciens agents utilisateurs qui ne prennent pas en charge [iframe](#) les éléments, le contenu serait analysé comme un balisage qui pourrait agir comme un contenu de secours.)

*L' [analyseur HTML](#) traite le balisage à l'intérieur [iframe](#) des éléments comme du texte.*



Les attributs IDL **src**, **srcdoc**, **name**, **sandbox** et **allow** doivent [réfléter](#) les attributs de contenu respectifs du même nom.

Les [jetons pris en charge](#) pour [sandbox](#)'s [DOMTokenList](#) sont les valeurs autorisées définies dans l' [sandbox](#) attribut et prises en charge par l'agent utilisateur.

L' **allowFullscreen** attribut IDL doit [réfléter](#) l' [allowfullscreen](#) attribut content.



L' **referrerPolicy** attribut IDL doit [réfléter](#) l' [referrerpolicy](#) attribut content, [limité aux seules valeurs connues](#) .

L' **loading** attribut IDL doit [réfléter](#) l' [loading](#) attribut content, [limité aux seules valeurs connues](#) .



Les **contentDocument** étapes du getter consistent à retourner le [document de contenu](#) de [this](#) .



Les **contentWindow** étapes du getter consistent à renvoyer [cette fenêtre de contenu](#) .

Voici un exemple de page utilisant un [iframe](#) pour inclure la publicité d'un régime publicitaire :

```
<iframe
src="https://ads.example.com/?customerid=923513721&format=banne
r"
width="468" height="60"></iframe>
```

#### 4.8.6 L' **embed** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu intégré](#) .  
[Contenu interactif](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

##### Modèle de contenu :

[Rien](#) .

##### Omission de balise dans text/html :

Pas [de balise de fin](#) .

##### Attributs de contenu :

### Attributs globaux

src— Adresse de la ressource

type— Type de ressource intégrée

width— Dimensions horizontales

height— Dimension verticale

Tout autre attribut qui n'a pas d'espace de noms (voir prose).

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]

interface HTMLEmbedElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute USVString src;

    [CEReactions] attribute DOMString type;

    [CEReactions] attribute DOMString width;

    [CEReactions] attribute DOMString height;

    Document? getSVGDocument();

    // also has obsolete members

};
```

L' embed élément fournit un point d'intégration pour une application externe ou un contenu interactif.

L' src attribut donne l' URL de la ressource intégrée. L'attribut, s'il est présent, doit contenir une URL non vide valide potentiellement entourée d'espaces .

Si l' itemprop attribut est spécifié sur un embed élément, alors l' src attribut doit également être spécifié.

L' type attribut, s'il est présent, donne le type MIME par lequel le plugin à instancier est sélectionné. La valeur doit être une chaîne de type MIME valide . Si l' type attribut et l' src attribut sont tous deux présents, alors l' type attribut doit

spécifier le même type que les [métadonnées explicites Content-Type](#) de la ressource donnée par l' [src](#)attribut.

Tant que l'une des conditions suivantes se produit, tout [plug-in](#) instancié pour l'élément doit être supprimé et l' [embed](#)élément ne [représente](#) rien :

- L'élément n'a ni [src](#)attribut ni [type](#)attribut.
- L'élément a un ancêtre [d'élément multimédia](#) .
- L'élément a un [object](#)élément ancêtre qui n'affiche *pas* son [contenu de secours](#) .

Un [embed](#)élément est dit **potentiellement actif** lorsque les conditions suivantes sont toutes réunies simultanément :

- L'élément se trouve [dans un document](#) ou se trouvait [dans un document](#) la dernière fois que la [boucle d'événements](#) a atteint [l'étape 1](#) .
- Le [document de nœud](#) de l'élément est [entièrement actif](#) .
- L'élément a soit un [src](#)ensemble d'attributs, soit un [type](#)ensemble d'attributs (ou les deux).
- L' [src](#)attribut de l'élément est soit absent, soit sa valeur n'est pas la chaîne vide.
- L'élément n'est pas un descendant d'un [élément média](#) .
- L'élément n'est pas un descendant d'un [object](#)élément qui n'affiche pas son [contenu de secours](#) .
- L' élément est [en cours de rendu](#) , ou était [en cours de rendu](#) la dernière fois que la [boucle d' événements](#) a atteint [l' étape 1](#) .

Chaque fois qu'un [embed](#)élément qui n'était pas [potentiellement actif](#) devient [potentiellement actif](#) , et chaque fois qu'un élément [potentiellement actif](#) [embed](#) qui reste [potentiellement actif](#) et dont [src](#)l'attribut est défini, modifié ou supprimé ou son [type](#)attribut défini, modifié ou supprimé, l'agent utilisateur doit [mettre un élément en file d'attente tâche](#) sur la **source de tâche d'intégration** donnée à l'élément pour exécuter [les embedétapes de configuration de l'élément](#) pour cet élément.

**Les [embed](#)étapes de configuration de l'élément** pour un *élément*[embed](#) élément donné sont les suivantes :

1. Si une autre [tâche](#) a depuis été mise en file d'attente pour exécuter [les embedétapes de configuration d'élément](#) pour *element* , alors revenez.
2. Si *l'élément* a un [src](#)ensemble d'attributs, alors :
  1. Soit *url* le résultat de [l'analyse](#) de la valeur de l'attribut de l' *élément*[src](#) , par rapport au [nœud document](#) de l' *élément* .
  2. Si *l'url* est un échec, alors revenez.

3. Soit *request* une nouvelle requête dont l'URL est *url* , le client est l'objet de paramètres pertinent du document de nœud de l'élément , la destination est " " , le mode d'identification est " " , le mode est " " , le type d'initiateur est " " et dont l' utilisation- L'indicateur d'informations d'identification d'URL est défini.`embedincludenavigateembed`
4. Récupérez la requête , avec processResponse défini sur les étapes suivantes en fonction de la réponse :
  1. Si une autre tâche a depuis été mise en file d'attente pour exécuter les `embed` étapes de configuration d'élément pour *element* , alors revenez.
  2. Si la réponse est une erreur réseau , déclenchez un événement nommé `load` à *element* et retournez.
  3. Soit *type* le résultat de la détermination du type de contenu donné *element* et *response* .
  4. Allumer le *type* :

nul

1. N'affiche aucun plugin pour l'élément .

**Sinon**

2. Si le contenu navigable de l' élément est nul, créez un nouvel élément navigable enfant pour l'élément .
3. Naviguez dans le contenu de l' élément navigable jusqu'à l'URL de la réponse en utilisant le document de nœud de l' élément , avec la réponse définie sur *response* et historyHandling défini sur " ".`replace`
4. L'élément représente maintenant son contenu navigable .

*L'attribut de l' élément `src` n'est pas mis à jour si le contenu navigable est ensuite navigué vers d'autres emplacements.*

- La récupération de la ressource doit retarder l'événement de chargement du document de nœud de l' élément .
3. La récupération de la ressource doit retarder l'événement de chargement du document de nœud de l' élément .
4. Sinon, n'affichez aucun plugin pour l'élément .

Pour déterminer le **type de contenu** donné à un élément`embed` *element* et à une réponse response , exécutez les étapes suivantes :



1. Si l'élément a un [type](#) attribut et que la valeur de cet attribut est un type pris en charge par un [plugin](#) , alors renvoyez la valeur de l' [type](#) attribut.
2. Si le composant de [chemin de l'URL](#) de la réponse correspond à un modèle pris en charge par un [plug-in](#) , renvoyez le type que ce plug-in peut gérer.

Par exemple, un plugin peut dire qu'il peut gérer les URL avec des composants [de chemin](#) qui se terminent par la chaîne de quatre caractères ".swf".

3. Si la réponse a des [métadonnées Content-Type explicites](#) et que cette valeur est un type pris en charge par un [plug-in](#) , renvoyez cette valeur.
4. Renvoie nul.

*Il est intentionnel que l'algorithme ci-dessus permette à la réponse d'avoir un [statut non ok](#) . Cela permet aux serveurs de renvoyer des données pour les plugins même avec des réponses d'erreur (par exemple, les codes d'erreur de serveur interne HTTP 500 peuvent toujours contenir des données de plugin).*

Pour **n'afficher aucun plugin** pour un élément [embed](#) element :

1. Détruit un élément [enfant navigable](#) donné .
2. Affiche une indication qu'aucun [plugin](#) n'a pu être trouvé pour *element* , comme le contenu de *element* .
3. l'élément ne [représente](#) plus rien.

*L' [embed](#) élément n'a pas [de contenu de secours](#) ; ses descendants sont ignorés.*

Chaque fois qu'un [embed](#) élément qui était [potentiellement actif](#) cesse d'être [potentiellement actif](#) , tout [plugin](#) qui avait été instancié pour cet élément doit être déchargé.

L' [embed](#) élément [retarde potentiellement l'événement load](#) .

L' [embed](#) élément prend en charge [les attributs de dimension](#) .

Les attributs IDL **src** et **type** chacun doivent [réfléter](#) les attributs de contenu respectifs du même nom.

#### 4.8.7 L' **object** élément



## Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu intégré](#) .  
[Élément associé au formulaire listé](#) .  
[Contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

## Modèle de contenu :

[Transparente](#) .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

### Attributs globaux

[data](#)— Adresse de la ressource  
[type](#)— Type de ressource intégrée  
[name](#)— Nom du [contenu navigable](#)  
[form](#)— Associe l'élément à un [form](#)élément  
[width](#)— Dimensions horizontales  
[height](#)— Dimension verticale

## Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

## Interface DOM :

[Exposed=Window]

```
interface HTMLObjectElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute USVString data;
```

```
    [CEReactions] attribute DOMString type;
```

```
    [CEReactions] attribute DOMString name;
```

```
    readonly attribute HTMLFormElement? form;
```

```
    [CEReactions] attribute DOMString width;
```

```

[CEReactions] attribute DOMString height;

readonly attribute Document? contentDocument;

readonly attribute WindowProxy? contentWindow;

Document? getSVGDocument();

readonly attribute boolean willValidate;

readonly attribute ValidityState validity;

readonly attribute DOMString validationMessage;

boolean checkValidity();

boolean reportValidity();

undefined setCustomValidity(DOMString error);

// also has obsolete members

};

```

Selon le type de contenu instancié par l' [object](#) élément, le nœud prend également en charge d'autres interfaces.

L' [object](#) élément peut représenter une ressource externe qui, selon le type de ressource, sera soit traitée comme une image, soit comme un [enfant navigable](#) .

L' [data](#) attribut spécifie l' [URL](#) de la ressource. Elle doit être présente et doit contenir une [URL non vide valide potentiellement entourée d'espaces](#) .

L' [type](#) attribut, s'il est présent, spécifie le type de la ressource. S'il est présent, l'attribut doit être une [chaîne de type MIME valide](#) .

L' [name](#) attribut, s'il est présent, doit être un [nom de cible navigable valide](#) . La valeur donnée est utilisée pour nommer le [contenu navigable](#) de l'élément , le cas échéant, et s'il est présent lors de [la création du contenu navigable](#) de l'élément .

Chaque fois que l'une des conditions suivantes se produit :

- l'élément est créé,

- l'élément est extrait de la [pile d'éléments ouverts](#) d'un [analyseur HTML](#) ou d'un [analyseur XML](#) ,
- l'élément n'est pas sur la [pile d'éléments ouverts](#) d'un [analyseur HTML](#) ou d'un [analyseur XML](#) , et il est soit [inséré dans un document](#) , soit [supprimé d'un document](#) ,
- le [document de nœud](#) de l'élément change s'il est [entièrement actif](#) ,
- l'un des [object](#)éléments ancêtres de l'élément passe à ou cesse d'afficher son [contenu de secours](#) ,
- l' [classid](#)attribut de l'élément est défini, modifié ou supprimé,
- l' [classid](#)attribut de l'élément n'est pas présent et son [data](#)attribut est défini, modifié ou supprimé,
- ni l' [classid](#)attribut de l'élément ni son [data](#)attribut ne sont présents, et son [type](#)attribut est défini, modifié ou supprimé,
- l'élément passe d' [être rendu](#) à ne pas être rendu, ou vice versa,

... l'agent utilisateur doit [mettre en file d'attente une tâche d'élément](#) sur la [source de la tâche de manipulation DOM](#) étant donné l' [object](#)élément pour exécuter les étapes suivantes pour (re) déterminer ce que l' [object](#)élément représente. Cette [tâche](#) en [file d'attente](#) ou en cours d'exécution doit [retarder l'événement load](#) du [nœud document](#) de l'élément .

1. Si l'utilisateur a indiqué une préférence pour que [le contenu de secours](#)[object](#) de cet élément soit affiché au lieu du comportement habituel de l'élément, passez à l'étape ci-dessous intitulée *fallback* .

*[Par exemple, un utilisateur peut demander que le contenu de secours de l'élément soit affiché car ce contenu utilise un format que l'utilisateur trouve plus accessible.](#)*

2. Si l'élément a un [élément média](#) ancêtre , ou a un élément ancêtre [object](#)qui n'affiche pas son [contenu de secours](#) , ou si l'élément n'est pas [dans un document](#) dont [le contexte de navigation](#) n'est pas nul, ou si le [nœud document](#) de l'élément n'est pas [entièrement actif](#) , ou si l'élément est toujours dans la [pile d'éléments ouverts](#) d'un [analyseur HTML](#) ou d'un [analyseur XML](#) , ou si l'élément n'est pas [en cours de rendu](#) , passez à l'étape ci-dessous intitulée *fallback* .
3. Si l' [data](#)attribut est présent et que sa valeur n'est pas la chaîne vide, alors :
  1. Si l' [type](#)attribut est présent et que sa valeur n'est pas un type pris en charge par l'agent utilisateur, alors l'agent utilisateur peut passer à l'étape ci-dessous intitulée *fallback* sans récupérer le contenu pour examiner son type réel.

2. [Analyser une URL](#) en fonction de l' [data](#) attribut, par rapport au [nœud document](#) de l'élément .
3. Si cela échoue, [déclenchez un événement](#) nommé [error](#) à l'élément, puis passez à l'étape ci-dessous intitulée *fallback* .
4. Soit *request* une nouvelle [requête](#) dont l'[URL](#) est l' [enregistrement d'URL résultant](#) , le [client](#) est l' [objet de paramètres pertinent](#) du [document de nœud](#) de l'élément , le [la destination](#) est " " , le [mode d'identification](#) est " " , le [mode](#) est " " , le [type d'initiateur](#) est " " et dont l'[indicateur use-URL-credentials](#) est défini.`objectincludenavigateobject`
5. [Récupérer](#) la *requête* .

La récupération de la ressource doit [retarder l'événement de chargement](#) du [document de nœud](#) de l'élément jusqu'à ce que la [tâche](#) qui est [mise en file](#) d'attente par la [source de tâche réseau](#) une fois que la ressource a été récupérée (définie ensuite) a été exécutée.

6. Si la ressource n'est pas encore disponible (par exemple parce que la ressource n'était pas disponible dans le cache, de sorte que le chargement de la ressource nécessitait de faire une demande sur le réseau), passez à l'étape ci-dessous intitulée *fallback* . La [tâche](#) qui est [mise en file d'attente](#) par la [source de tâches réseau](#) une fois que la ressource est disponible doit redémarrer cet algorithme à partir de cette étape. Les ressources peuvent se charger de manière incrémentielle ; les agents utilisateurs peuvent choisir de considérer une ressource "disponible" chaque fois que suffisamment de données ont été obtenues pour commencer à traiter la ressource.
7. Si le chargement a échoué (par exemple, il y a eu une erreur HTTP 404, il y a eu une erreur DNS), [déclenchez un événement](#) nommé [error](#) au niveau de l'élément, puis passez à l'étape ci-dessous intitulée *fallback* .
8. Déterminez le *type de ressource* , comme suit :
  1. Soit le *type de ressource* inconnu.
  2. Si l'agent utilisateur est configuré pour obéir strictement aux en-têtes Content-Type pour cette ressource et que la ressource a des [métadonnées Content-Type associées](#) , laissez le *type de ressource* être le type spécifié dans [les métadonnées Content-Type de la ressource](#) et passez à l'étape ci-dessous *gestionnaire* étiqueté .

**Cela peut introduire une vulnérabilité, dans laquelle un site tente d'intégrer une ressource qui utilise un type particulier, mais le site distant remplace cela et fournit à la place à l'agent utilisateur une ressource qui déclenche un type de**

**contenu différent avec des caractéristiques de sécurité différentes.**

3. Exécutez l'ensemble d'étapes approprié dans la liste suivante :

**Si la ressource a des métadonnées Content-Type associées**

1. Soit *binaire* faux.
2. Si le type spécifié dans les métadonnées Content-Type de la ressource est "text/plain", et que le résultat de l'application des règles permettant de distinguer si une ressource est textuelle ou binaire à la ressource est que la ressource n'est pas text/plain, alors définissez *binary* sur true.
3. Si le type spécifié dans les métadonnées Content-Type de la ressource est "application/octet-stream", alors définissez *binary* sur true.
4. Si *binary* est false, laissez le *type de ressource* être le type spécifié dans les métadonnées Content-Type de la ressource et passez à l'étape ci-dessous intitulée *handler*.
5. Si un typeattribut est présent sur l' objectélément et que sa valeur n'est pas application/octet-stream, exécutez les étapes suivantes :
  1. Si la valeur de l'attribut est un type commençant par "image/" qui n'est pas également un type XML MIME, laissez le *type de ressource* être le type spécifié dans cet typeattribut.
  2. Passez à l'étape ci-dessous intitulée *handler*.

**Sinon, si la ressource n'a pas de métadonnées Content-Type associées**

6. Si un attribut est typeprésent sur l' objectélément, laissez le *type provisoire* être le type spécifié dans cet typeattribut.

Sinon, laissez *type provisoire* être le type calculé de la ressource.

7. Si le *type provisoire* n'est pas application/octet-stream, alors laissez le *type de ressource* être *type provisoire* et passez à l'étape sous le *gestionnaire* étiqueté .
  4. Si l'application de l' algorithme d'analyseur d'URL à l' URL de la ressource spécifiée (après toute redirection) aboutit à un enregistrement d'URL dont le composant de chemin correspond à un modèle pris en charge par un plug-in, laissez le *type de ressource* être le type que ce plug-in peut gérer.

Par exemple, un plugin peut dire qu'il peut gérer des ressources avec des composants [de chemin](#) qui se terminent par la chaîne de quatre caractères ".swf".

9. *Il est possible que cette étape se termine ou que l'une des sous-étapes ci-dessus passe directement à l'étape suivante, le type de ressource étant toujours inconnu. Dans les deux cas, l'étape suivante déclenchera le repli.*

10. *Handler* : traite le contenu comme indiqué par le premier des cas suivants qui correspond :

**Si le type de ressource est un [type XML MIME](#) , ou si le type de ressource ne commence pas par " [image/](#) "**

Si le [contenu navigable](#)[object](#) de l'élément est nul, [créez un nouveau navigable enfant](#) pour l'élément.

Soit *response* la [réponse](#) de [fetch](#) .

Si l' [URL](#) de la *réponse* ne [correspond](#) pas , alors [naviguez](#) dans le contenu de l'élément [navigable](#) vers l' [URL](#) de la *réponse* en utilisant le [noeud document](#) de l'élément , avec [historyHandling](#) défini sur " [.about:blank](#)[replace](#) "

*L' [data](#)attribut de l' [object](#)élément n'est pas mis à jour si le [contenu navigable](#) est ensuite [navigué](#) vers d'autres emplacements.*

L' [object](#)élément [représente](#) son [contenu navigable](#) .

**Si le type de ressource commence par " [image/](#) " et que la prise en charge des images n'a pas été désactivée**

[Détruire un enfant navigable](#) compte tenu de l' [object](#)élément.

Appliquez les règles [de reniflage d'image](#) pour déterminer le type de l'image.

L' [object](#)élément [représente](#) l'image spécifiée.

Si l'image ne peut pas être rendue, par exemple parce qu'elle est malformée ou dans un format non pris en charge, passez à l'étape ci-dessous intitulée *fallback* .

### **Sinon**

Le *type de ressource* donné n'est pas pris en charge. Passez à l'étape ci-dessous intitulée *fallback* .

*Si l'étape précédente s'est terminée avec le type de ressource inconnu, c'est le cas qui est déclenché.*

11. Le contenu de l'élément ne fait pas partie de ce que l' [object](#)élément représente.

12. Si l' object élément ne représente pas son contenu navigable , une fois la ressource complètement chargée, mettez en file d'attente une tâche d'élément sur la source de la tâche de manipulation DOM donnée à l' object élément pour déclencher un événement nommé load sur l'élément.

*Si l'élément représente son contenu navigable , alors une tâche analogue sera mise en file d'attente lorsque la création Document aura complètement fini de se charger .*

13. Retour.

4. *Fallback* : L' object élément représente les enfants de l'élément. Il s'agit du contenu de secours de l'élément . Détruire un enfant navigable compte tenu de l'élément.

En raison de l'algorithme ci-dessus, le contenu des object éléments agit comme un contenu de secours , utilisé uniquement lorsque les ressources référencées ne peuvent pas être affichées (par exemple, parce qu'elles ont renvoyé une erreur 404). Cela permet à plusieurs object éléments d'être imbriqués les uns dans les autres, ciblant plusieurs agents utilisateurs avec des capacités différentes, l'agent utilisateur choisissant le premier qu'il prend en charge.

L' object élément retarde potentiellement l'événement load .

L' form attribut est utilisé pour associer explicitement l' object élément à son propriétaire de formulaire .

L' object élément prend en charge les attributs de dimension .



Les attributs IDL data, type et name chacun doivent refléter les attributs de contenu respectifs du même nom.



Les contentDocument étapes du getter consistent à retourner ce document de contenu .



Les contentWindow étapes du getter consistent à renvoyer cette fenêtre de contenu .

Les attributs , et et les willValidate méthodes validity, et font partie de l' API de validation de contrainte . L' attribut IDL fait partie de l'API des formulaires de l'élément. validationMessage checkValidity() reportValidity() setCustomValidity() form



Dans cet exemple, une page HTML est imbriquée dans une autre à l'aide de l'[object](#) élément .

```
<figure>
  <object data="clock.html"></object>
  <figcaption>My HTML Clock</figcaption>
</figure>
```

#### 4.8.8 L' [video](#)élément



##### Catégories :

[Contenu du flux](#) .

[Contenu de la phrase](#) .

[Contenu intégré](#) .

Si l'élément a un [controls](#)attribut : [Contenu interactif](#) .

[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

##### Modèle de contenu :

Si l'élément a un [src](#)attribut : zéro ou plusieurs [track](#)éléments, alors [transparent](#) , mais sans descendants [d'éléments multimédias](#) .

Si l'élément n'a pas d' [src](#)attribut : zéro ou plusieurs [source](#)éléments, puis zéro ou plusieurs [track](#)éléments, puis [transparent](#) , mais sans descendants [d'éléments multimédias](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

###### Attributs globaux

[src](#)— Adresse de la ressource

[crossorigin](#)— Comment l'élément gère les requêtes crossorigin

[poster](#)— Cadre d'affiche à afficher avant la lecture de la vidéo

[preload](#)- Indique la quantité de mémoire tampon dont la [ressource multimédia](#) aura probablement besoin

[autoplay](#)— Indique que la [ressource multimédia](#) peut être démarrée automatiquement lorsque la page est chargée

[playsinline](#)— Encouragez l'agent utilisateur à afficher le contenu vidéo dans la zone de lecture de l'élément

[loop](#)— S'il faut boucler la [ressource multimédia](#)

[muted](#)— S'il faut désactiver la [ressource multimédia](#) par défaut

controls— Afficher les commandes de l'agent utilisateur  
width— Dimensions horizontales  
height— Dimension verticale

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]

interface HTMLVideoElement : HTMLMediaElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute unsigned long width;

    [CEReactions] attribute unsigned long height;

    readonly attribute unsigned long videoWidth;

    readonly attribute unsigned long videoHeight;

    [CEReactions] attribute USVString poster;

    [CEReactions] attribute boolean playsInline;

};
```

Un videoélément est utilisé pour lire des vidéos ou des films, et des fichiers audio avec des légendes.

Le contenu peut être fourni à l'intérieur de l' videoélément. Les agents utilisateurs ne doivent pas montrer ce contenu à l'utilisateur ; il est destiné aux navigateurs Web plus anciens qui ne prennent pas en charge video, afin que les plug-ins vidéo hérités puissent être essayés, ou pour afficher du texte aux utilisateurs de ces navigateurs plus anciens les informant de la manière d'accéder au contenu vidéo.

*En particulier, ce contenu n'est pas destiné à répondre aux problèmes d'accessibilité. Pour rendre le contenu vidéo accessible aux malvoyants, aux aveugles, aux malentendants, aux sourds et aux personnes souffrant d'autres handicaps physiques ou cognitifs, diverses fonctionnalités sont disponibles. Des sous-titres peuvent être fournis, soit intégrés dans le flux vidéo, soit sous forme de fichiers externes à l'aide de l' trackélément. Des pistes en langue des signes peuvent être intégrées dans le flux vidéo. Les descriptions audio peuvent être*

intégrées dans le flux vidéo ou sous forme de texte à l'aide d'un [fichier WebVTT](#) référencé à l'aide du [track](#) élément et synthétisé en parole par l'agent utilisateur. WebVTT peut également être utilisé pour fournir des titres de chapitre. Pour les utilisateurs qui préfèrent ne pas utiliser un élément multimédia du tout, des transcriptions ou d'autres alternatives textuelles peuvent être fournies en les reliant simplement dans la prose près de l' [video](#) élément. [\[WEBVTT\]](#)

L' [video](#) élément est un [élément multimédia](#) dont [les données multimédias](#) sont apparemment des données vidéo, éventuellement avec des données audio associées.

Les attributs [src](#), [crossorigin](#), [preload](#), [autoplay](#), [loop](#), [muted](#) et sont [les attributs communs à tous les éléments multimédias](#) .[controls](#)

L' [poster](#) attribut donne l' [URL](#) d'un fichier image que l'agent utilisateur peut afficher lorsqu'aucune donnée vidéo n'est disponible. L'attribut, s'il est présent, doit contenir une [URL non vide valide potentiellement entourée d'espaces](#) .

Si la ressource spécifiée doit être utilisée, alors, lorsque l'élément est créé ou lorsque l' [poster](#) attribut est défini, modifié ou supprimé, l'agent utilisateur doit exécuter les étapes suivantes pour déterminer le **poster** de l'élément (quelle que soit la valeur de l'élément [afficher le drapeau de l'affiche](#) ):

1. S'il existe une instance existante de cet algorithme en cours d'exécution pour cet [video](#) élément, abandonnez cette instance de cet algorithme sans modifier l' [affiche](#) .
2. Si la [poster](#) valeur de l'attribut est la chaîne vide ou si l'attribut est absent, alors il n'y a pas [de poster](#) ; retour.
3. [Analyse](#) la [poster](#) valeur de l'attribut par rapport au [nœud document](#) de l'élément . Si cela échoue, alors il n'y a pas [de cadre d'affiche](#) ; retour.
4. Soit *request* une nouvelle [requête](#) dont l'[URL](#) est l' [enregistrement d'URL résultant](#) , [le client](#) est l' [objet de paramètres pertinent](#) du [document de nœud](#) de l'élément , [la destination](#) est " " , [le type d'initiateur](#) est " " , [le mode d'identification](#) est " " et dont [use-URL-credentials drapeau](#) est défini. `imagevideoinclude`
5. [Récupérer](#) la *requête* . Cela doit [retarder l'événement load](#) du [nœud document](#) de l'élément .
6. Si une image est ainsi obtenue, le [cadre de l'affiche](#) est cette image. Sinon, il n'y a pas [de cadre d'affiche](#) .

L'image donnée par l' [poster](#) attribut, le [poster frame](#) , est censée être une image représentative de la vidéo (généralement l'une des premières images non vierges) qui donne à l'utilisateur une idée de ce à quoi ressemble la vidéo.

L' **playsinline**attribut est un [attribut booléen](#) . S'il est présent, il sert d'indice à l'agent utilisateur que la vidéo doit être affichée "en ligne" dans le document par défaut, contrainte à la zone de lecture de l'élément, au lieu d'être affichée en plein écran ou dans une fenêtre redimensionnable indépendante.

*L'absence de l' **playsinline** attribut n'implique pas que la vidéo s'affichera en plein écran par défaut. En effet, la plupart des agents utilisateurs ont choisi de lire toutes les vidéos en ligne par défaut, et dans ces agents utilisateurs, l' **playsinline**attribut n'a aucun effet.*

---

Un **video**élément représente ce qui est donné pour la première condition correspondante dans la liste ci-dessous :

Lorsqu'aucune donnée vidéo n'est disponible (l' **readyState**attribut de l'élément est **HAVE NOTHING**, ou **HAVE METADATA**mais aucune donnée vidéo n'a encore été obtenue, ou l' **readyState**attribut de l'élément est une valeur ultérieure mais la **ressource multimédia** n'a pas de canal vidéo)

L' **video**élément [représente](#) son [cadre d'affiche](#) , le cas échéant, ou bien [un noir transparent](#) sans [dimensions intrinsèques](#) .

Lorsque l' **video**élément est [en pause](#) , la [position de lecture actuelle](#) est la première image de la vidéo et l' [indicateur d'affichage](#) de l'élément est défini

L' **video**élément [représente](#) son [affiche](#) , le cas échéant, ou bien la première image de la vidéo.

Lorsque l' **video**élément est [en pause](#) et que l'image de la vidéo correspondant à la [position de lecture actuelle](#) n'est pas disponible (par exemple, parce que la vidéo est en recherche ou mise en mémoire tampon)

Lorsque l' **video**élément n'est ni [potentiellement en cours de lecture](#) ni [en pause](#) (par exemple, lors d'une recherche ou d'un blocage)

L' **video**élément [représente](#) la dernière image de la vidéo à avoir été rendue.

Lorsque l' **video**élément est [en pause](#)

L' **video**élément [représente](#) l'image de la vidéo correspondant à la [position de lecture actuelle](#) .

Sinon (l' **video**élément a un canal vidéo et est [potentiellement en cours de lecture](#) )

L' **video**élément [représente](#) l'image de la vidéo à la [position "actuelle"](#) en augmentation continue . Lorsque la [position de lecture actuelle](#) change de sorte que la dernière image rendue n'est plus l'image correspondant à la [position de lecture actuelle](#) dans la vidéo, la nouvelle image doit être rendue.

Les images vidéo doivent être obtenues à partir de la piste vidéo qui a été [sélectionnée](#) lorsque la [boucle d'événements](#) a atteint [l'étape 1](#) pour la dernière fois .

*Quelle image dans un flux vidéo correspond à une position de lecture particulière est définie par le format du flux vidéo.*

L'élément [représente](#)[video](#) également tous [les repères de piste de texte](#) dont [l'indicateur actif de repère de piste de texte](#) est défini et dont [la piste de texte](#) est en mode [d'affichage](#) , et tout audio de la [ressource multimédia](#) , à la [position de lecture actuelle](#) .

Tout audio associé à la [ressource multimédia](#) doit, s'il est lu, être lu synchronisé avec la [position de lecture actuelle](#) , au [volume multimédia effectif](#) de l'élément . L'agent utilisateur doit lire l'audio des pistes audio qui ont été [activées](#) lorsque la [boucle d'événements](#) a atteint l'étape 1 pour la dernière fois.

En plus de ce qui précède, l'agent utilisateur peut fournir des messages à l'utilisateur (tels que "mise en mémoire tampon", "aucune vidéo chargée", "erreur" ou des informations plus détaillées) en superposant du texte ou des icônes sur la vidéo ou d'autres zones du zone de lecture de l'élément, ou d'une autre manière appropriée.

Les agents utilisateurs qui ne peuvent pas restituer la vidéo peuvent à la place faire en sorte que l'élément [représente](#) un lien vers un utilitaire de lecture vidéo externe ou vers les données vidéo elles-mêmes.

Lorsque la [ressource multimédia](#)[video](#) d'un élément possède un canal vidéo, l'élément fournit une [source de peinture](#) dont la largeur est la [largeur intrinsèque](#) de la [ressource multimédia](#) , dont la hauteur est la [hauteur intrinsèque](#) de la [ressource multimédia](#) et dont l'apparence est l'image de la vidéo correspondant à [la position de lecture](#) , si elle est disponible, ou bien (par exemple, lorsque la vidéo est en recherche ou mise en mémoire tampon) son apparence précédente, le cas échéant, ou bien (par exemple, parce que la vidéo charge toujours la première image) la noirceur.

---

`video.videoWidth`

✓ 

`video.videoHeight`

✓ 

Ces attributs renvoient les dimensions intrinsèques de la vidéo, ou zéro si les dimensions ne sont pas connues.

La **largeur intrinsèque** et la **hauteur intrinsèque** de la [ressource multimédia](#) sont les dimensions de la ressource en [pixels CSS](#) après avoir pris en compte les dimensions de la ressource, le format d'image, l'ouverture propre, la résolution, etc., tels que définis pour le format utilisé par la ressource. Si un format anamorphique ne définit pas comment appliquer le rapport d'aspect aux dimensions des données vidéo pour obtenir les dimensions "correctes", alors l'agent utilisateur doit appliquer le rapport en augmentant une dimension et en laissant l'autre inchangée.

L' **videoWidth**attribut IDL doit renvoyer la [largeur intrinsèque](#) de la vidéo en [pixels CSS](#) . L' **videoHeight**attribut IDL doit renvoyer la [hauteur intrinsèque](#) de la vidéo en [pixels CSS](#) . Si l' **readyState**attribut de l'élément est [HAVE NOTHING](#), alors les attributs doivent retourner 0.

Chaque fois que la [largeur intrinsèque](#) ou la [hauteur intrinsèque](#) de la vidéo change (y compris, par exemple, parce que la [piste vidéo sélectionnée](#) a été modifiée), si l'attribut de l'élément **readyState** n'est pas [HAVE NOTHING](#), l'agent utilisateur doit [mettre en file d'attente une tâche d'élément multimédia](#) étant donné que l' [élément multimédia](#) déclenche [un événement](#) nommé [resize](#) au niveau de l' [élément média](#) .

L' **video**élément prend en charge [les attributs de dimension](#) .

En l'absence de règles de style contraires, le contenu vidéo doit être rendu à l'intérieur de la zone de lecture de l'élément de sorte que le contenu vidéo soit affiché centré dans la zone de lecture à la plus grande taille possible qui s'y adapte complètement, le rapport d'aspect du contenu vidéo étant conservé. Ainsi, si le rapport d'aspect de la zone de lecture ne correspond pas au rapport d'aspect de la vidéo, la vidéo sera affichée en format letterbox ou pillarbox. Les zones de la zone de lecture de l'élément qui ne contiennent pas la vidéo ne représentent rien.

*Dans les agents utilisateurs qui implémentent CSS, l'exigence ci-dessus peut être implémentée en utilisant la [règle de style suggérée dans la section Rendu](#) .*

La [largeur intrinsèque](#) de la **video**zone de lecture d'un élément est la [largeur intrinsèque](#) du [poster frame](#) , si celle-ci est disponible et que l'élément [représente](#) actuellement son poster frame ; sinon, c'est la [largeur intrinsèque](#) de la ressource vidéo, si celle-ci est disponible ; sinon la [largeur intrinsèque](#) est manquante.

La [hauteur intrinsèque](#) de la **video**zone de lecture d'un élément est la [hauteur intrinsèque](#) du [poster frame](#) , si celle-ci est disponible et que l'élément [représente](#) actuellement son poster frame ; sinon c'est la [hauteur intrinsèque](#) de la ressource vidéo, si celle-ci est disponible ; sinon la [hauteur intrinsèque](#) est manquante.

La [taille d'objet par défaut](#) est une largeur de 300 [pixels CSS](#) et une hauteur de 150 [pixels CSS](#) . [\[CSSIMAGES\]](#)

---

Les agents utilisateurs doivent fournir des contrôles pour activer ou désactiver l'affichage des sous-titres, des pistes de description audio et d'autres données supplémentaires associées au flux vidéo, bien que ces fonctionnalités ne doivent, encore une fois, pas interférer avec le rendu normal de la page.

Les agents utilisateurs peuvent permettre aux utilisateurs de visualiser le contenu vidéo d'une manière plus adaptée à l'utilisateur, comme en plein écran ou dans une fenêtre redimensionnable indépendante. Les agents utilisateurs peuvent même déclencher un tel mode de visualisation par défaut lors de la lecture d'une vidéo, bien qu'ils ne devraient pas le faire lorsque l'`playsinline` attribut est spécifié. Comme pour les autres fonctionnalités de l'interface utilisateur, les contrôles permettant de l'activer ne doivent pas interférer avec le rendu normal de la page, sauf si l'agent utilisateur expose [une interface utilisateur](#). Dans un tel mode de visualisation indépendant, cependant, les agents utilisateurs peuvent rendre visibles des interfaces utilisateur complètes, même si l'`controls` attribut est absent.

Les agents utilisateurs peuvent autoriser la lecture vidéo à affecter les fonctionnalités du système susceptibles d'interférer avec l'expérience de l'utilisateur ; par exemple, les agents utilisateurs pourraient désactiver les économiseurs d'écran pendant que la lecture vidéo est en cours.

---

L'`poster` attribut IDL doit [réfléter](#) l'`poster` attribut content.

L'`playsInline` attribut IDL doit [réfléter](#) l'`playsinline` attribut content.

Cet exemple montre comment détecter lorsqu'une vidéo n'a pas pu être lue correctement :

```
<script>
function failed(e) {
    // video playback failed - show a message saying why
    switch (e.target.error.code) {
        case e.target.error.MEDIA_ERR_ABORTED:
            alert('You aborted the video playback.');
            break;
        case e.target.error.MEDIA_ERR_NETWORK:
            alert('A network error caused the video download to fail
part-way.');
            break;
```

```

        case e.target.error.MEDIA_ERR_DECODE:
            alert('The video playback was aborted due to a corruption
problem or because the video used features your browser did not
support.');
```

```

            break;

        case e.target.error.MEDIA_ERR_SRC_NOT_SUPPORTED:
            alert('The video could not be loaded, either because the
server or network failed or because the format is not supported.');
```

```

            break;

        default:
            alert('An unknown error occurred.');
```

```

            break;
    }
}
</script>

<p><video src="tgif.vid" autoplay controls
onerror="failed(event)"></video></p>

<p><a href="tgif.vid">Download the video file</a>.</p>
```

#### 4.8.9 L' **audio**élément



##### Catégories :

[Contenu du flux](#) .

[Contenu de la phrase](#) .

[Contenu intégré](#) .

Si l'élément a un [controls](#)attribut : [Contenu interactif](#) .

Si l'élément a un [controls](#)attribut : [Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

##### Modèle de contenu :

Si l'élément a un [src](#)attribut : zéro ou plusieurs [track](#)éléments, alors [transparent](#) , mais sans descendants [d'éléments multimédias](#) .

Si l'élément n'a pas d' [src](#)attribut : zéro ou plusieurs [source](#)éléments, puis zéro ou plusieurs [track](#)éléments, puis [transparent](#) , mais sans descendants [d'éléments multimédias](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :



### Attributs globaux

src— Adresse de la ressource

crossorigin— Comment l'élément gère les requêtes crossorigin

preload— Indique la quantité de mémoire tampon dont la ressource multimédia aura probablement besoin

autoplay— Indique que la ressource multimédia peut être démarrée automatiquement lorsque la page est chargée

loop— S'il faut boucler la ressource multimédia

muted— S'il faut désactiver la ressource multimédia par défaut

controls— Afficher les commandes de l'agent utilisateur

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window,
```

```
LegacyFactoryFunction=Audio(optional DOMString src)]
```

```
interface HTMLAudioElement : HTMLMediaElement {
```

```
[HTMLConstructor] constructor();
```

```
};
```

Un audioélément représente un son ou un flux audio.

Le contenu peut être fourni à l'intérieur de l' audioélément. Les agents utilisateurs ne doivent pas montrer ce contenu à l'utilisateur ; il est destiné aux navigateurs Web plus anciens qui ne prennent pas en charge audio, afin que les plug-ins audio hérités puissent être essayés, ou pour afficher du texte aux utilisateurs de ces navigateurs plus anciens les informant de la manière d'accéder au contenu audio.

*En particulier, ce contenu n'est pas destiné à répondre aux problèmes d'accessibilité. Pour rendre le contenu audio accessible aux sourds ou aux personnes souffrant d'autres handicaps physiques ou cognitifs, une variété de fonctionnalités sont disponibles. Si des sous-titres ou une vidéo en langue des signes sont disponibles, l' videoélément peut être utilisé à la place de l' audioélément pour lire l'audio, permettant aux utilisateurs d'activer les alternatives visuelles. Des titres de chapitre peuvent être fournis pour faciliter la navigation, en utilisant l' trackélément et un fichier WebVTT . Et, naturellement, des transcriptions ou d'autres alternatives textuelles peuvent être fournies en les reliant simplement dans la prose près de l' audioélément. [WEBVTT]*

L' audioélément est un élément multimédia dont les données multimédias sont apparemment des données audio.

Les attributs [src](#), [crossorigin](#), [preload](#), [autoplay](#), [loop](#), [muted](#) et [controls](#) sont [les attributs communs à tous les éléments multimédias](#).

```
audio = new Audio([ url ])
```

✓

Renvoie un nouvel [audio](#) élément, avec l' [src](#) attribut défini sur la valeur passée dans l'argument, le cas échéant.

Une fonction de fabrique héritée est fournie pour créer [HTMLAudioElement](#) des objets (en plus des méthodes de fabrique du DOM telles que [createElement\(\)](#)): . Lorsqu'elle est invoquée, la fonction de fabrique héritée doit effectuer les étapes suivantes :[Audio \(src\)](#)

1. Soit *document* l' objet [global actuel associé](#)[Document](#) .
2. Soit *audio* le résultat de [la création d'un élément](#) donné *document* , [audio](#) et l' [espace de noms HTML](#) .
3. [Définissez une valeur d'attribut](#) pour l'*audio* en utilisant " [preload](#)" et " [auto](#)".
4. Si *src* est donné, [définissez une valeur d'attribut](#) pour l'*audio* en utilisant " [src](#)" et *src* . (Cela [obligera l'agent utilisateur à invoquer](#) l' [algorithme de sélection des ressources](#) de l'objet avant de revenir.)
5. Retour *audio* .

#### 4.8.10 L' [track](#) élément

✓

MDN

✓

MDN

##### [Catégories](#) :

Aucun.

##### [Contextes dans lesquels cet élément peut être utilisé](#) :

En tant qu'enfant d'un [élément média](#) , avant tout [flux de contenu](#) .

##### [Modèle de contenu](#) :

[Rien](#) .

##### [Omission de balise dans text/html](#) :

Pas [de balise de fin](#) .

##### [Attributs de contenu](#) :

[Attributs globaux](#)

[kind](#) — Le type de piste de texte

src— Adresse de la ressource  
srclang— Langue de la piste de texte  
label— Étiquette visible par l'utilisateur  
default— Activez la piste si aucune autre [piste de texte](#) n'est plus appropriée

### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### Interface DOM :

[Exposed=Window]

interface **HTMLTrackElement** : [HTMLElement](#) {

[[HTMLConstructor](#)] constructor();

[[CEReactions](#)] attribute DOMString [kind](#);

[[CEReactions](#)] attribute USVString [src](#);

[[CEReactions](#)] attribute DOMString [srclang](#);

[[CEReactions](#)] attribute DOMString [label](#);

[[CEReactions](#)] attribute boolean [default](#);

const unsigned short [NONE](#) = 0;

const unsigned short [LOADING](#) = 1;

const unsigned short [LOADED](#) = 2;

const unsigned short [ERROR](#) = 3;

readonly attribute unsigned short [readyState](#);

readonly attribute [TextTrack](#) [track](#);

};

L' **track**élément permet aux auteurs de spécifier [des pistes de texte](#) temporisées externes explicites pour [les éléments multimédias](#) . Il ne [représente](#) rien en soi.

L' **kind**attribut est un [attribut énuméré](#) . Le tableau suivant répertorie les mots clés définis pour cet attribut. Le mot-clé indiqué dans la première cellule de chaque ligne correspond à l'état indiqué dans la deuxième cellule.

Mot-clé	État	Brève description
<b>subtitles</b>	<b>Les sous-titres</b>	Transcription ou traduction du dialogue, adaptée lorsque le son est disponible mais non compris (par exemple parce que l'utilisateur ne comprend pas la langue de la piste audio de la <a href="#">ressource multimédia</a> ). Superposé sur la vidéo.
<b>captions</b>	<b>Légendes</b>	Transcription ou traduction du dialogue, des effets sonores, des indices musicaux pertinents et d'autres informations audio pertinentes, adaptée lorsque le son n'est pas disponible ou n'est pas clairement audible (par exemple, parce qu'il est coupé, noyé par le bruit ambiant ou parce que l'utilisateur est sourd ). Superposé sur la vidéo ; marqué comme approprié pour les malentendants.
<b>descriptions</b>	<b>Descriptions</b>	Descriptions textuelles de la composante vidéo de la <a href="#">ressource multimédia</a> , destinées à la synthèse audio lorsque la composante visuelle est masquée, indisponible ou inutilisable (par exemple, parce que l'utilisateur interagit avec l'application sans écran en conduisant, ou parce que l'utilisateur est aveugle) . Synthétisé en tant qu'audio.
<b>chapters</b>	<b>Métadonnées des chapitres</b>	Pistes destinées à être utilisées à partir du script. Non affiché par l'agent utilisateur.
<b>metadata</b>	<b>Métadonnées</b>	

L'attribut peut être omis. La [valeur manquante par défaut](#) est l' état [des sous-titres](#) . La [valeur par défaut non valide](#) est l' état [des métadonnées](#) .

L' **src**attribut donne l' [URL](#) des données de piste de texte. La valeur doit être une [URL non vide valide potentiellement entourée d'espaces](#) . Cet attribut doit être présent.

Si l'élément a un **src**attribut dont la valeur n'est pas la chaîne vide et dont la valeur, lorsque l'attribut a été défini, a pu être [analysée](#) avec succès par rapport au [nœud document](#) de l'élément , alors l' **URL de suivi** de l'élément est la [chaîne d'URL résultante](#) . Sinon, l' [URL de piste](#) de l'élément est la chaîne vide.

Si l' [URL de suivi](#) de l'élément identifie une ressource WebVTT et que l' **kind**attribut de l'élément n'est pas dans l' état des métadonnées ou [des métadonnées du](#)

[chapitre](#) , le fichier WebVTT doit être un [fichier WebVTT utilisant un texte de repère](#) . [\[WEBVTT\]](#)

L' **srclang** attribut donne la langue des données de piste de texte. La valeur doit être une balise de langue BCP 47 valide. Cet attribut doit être présent si l' **kind** attribut de l'élément est dans l' état [des sous-titres](#) . [\[BCP47\]](#)

Si l'élément a un **srclang**attribut dont la valeur n'est pas la chaîne vide, alors la **langue de piste** de l'élément est la valeur de l'attribut. Sinon, l'élément n'a pas [de langue de piste](#) .

L' **label**attribut donne un titre lisible par l'utilisateur pour la piste. Ce titre est utilisé par les agents utilisateurs lorsqu'ils répertorient [les sous-titres](#) , [les légendes](#) et les pistes [de description audio](#) dans leur interface utilisateur.

La valeur de l' **label**attribut, si l'attribut est présent, ne doit pas être la chaîne vide. De plus, il ne doit pas y avoir deux **track** éléments enfants du même [élément média](#) dont **kind**les attributs sont dans le même état, dont **srclang**les attributs sont tous les deux manquants ou ont des valeurs qui représentent la même langue, et dont **label**les attributs sont à nouveau tous les deux manquants ou les deux ont la même valeur.

Si l'élément a un **label**attribut dont la valeur n'est pas la chaîne vide, alors l' **étiquette de piste** de l'élément est la valeur de l'attribut. Sinon, l' [étiquette de piste](#) de l'élément est une chaîne vide.

L' **default** attribut est un [attribut booléen](#) qui, s'il est spécifié, indique que la piste doit être activée si les préférences de l'utilisateur n'indiquent pas qu'une autre piste serait plus appropriée.

Chaque [élément multimédia](#) ne doit pas avoir plus d'un **track**élément enfant dont **kind**l'attribut est à l' état [sous-titres](#) ou [légendes](#) et dont **default**l'attribut est spécifié.

Chaque [élément média](#) ne doit pas avoir plus d'un **track**élément enfant dont **kind**l'attribut est dans l' état [description](#) et dont **default**l'attribut est spécifié.

Chaque [élément média](#) ne doit pas avoir plus d'un **track**enfant d'élément dont **kind**l'attribut est dans l' état [des métadonnées des chapitres](#) et dont **default**l'attribut est spécifié.

*Il n'y a pas de limite au nombre d' **track**éléments dont **kind**l'attribut est dans l' état [des métadonnées](#) et dont **default**l'attribut est spécifié.*

**track.readyState**

Renvoie l' [état de préparation de la piste de texte](#) , représenté par un nombre de la liste suivante :

**track.NONE** (0)

L' état [de la piste de texte n'est pas chargé](#) .  
`track.LOADING` (1)  
L' état [de chargement de la piste de texte](#) .  
`track.LOADED` (2)  
L' état [chargé de la piste de texte](#) .  
`track.ERROR` (3)  
La [piste de texte n'a pas réussi à charger](#) l'état.

`track.track`

Renvoie l' [TextTrack](#) objet correspondant à la [piste de texte](#) de l' [track](#) élément.

L' `readyState` attribut doit renvoyer la valeur numérique correspondant à l' [état de préparation de la piste de texte](#) de la [piste de texte](#) `track` de l'élément , tel que défini par la liste suivante :

**NONE**(valeur numérique 0)

L' état [de la piste de texte n'est pas chargé](#) .

**LOADING**(valeur numérique 1)

L' état [de chargement de la piste de texte](#) .

**LOADED**(valeur numérique 2)

L' état [chargé de la piste de texte](#) .

**ERROR**(valeur numérique 3)

La [piste de texte n'a pas réussi à charger](#) l'état.

L' `track` attribut IDL doit, lors de l'obtention, renvoyer l' objet correspondant de [la piste de texte](#) `track` de l'élément [TextTrack](#)



Les attributs `src`, `srcset`, et IDL doivent [refléter](#) `src` les attributs de contenu respectifs du même nom. L' attribut IDL doit [refléter](#) l'attribut de contenu du même nom, [limité aux seules valeurs connues](#) `labeldefaultkind`

Cette vidéo est sous-titrée en plusieurs langues :

```
<video src="brave.webm">
  <track kind=subtitles src=brave.en.vtt srclang=en label="English">
  <track kind=captions src=brave.en.hoh.vtt srclang=en
label="English for the Hard of Hearing">
  <track kind=subtitles src=brave.fr.vtt srclang=fr lang=fr
label="Français">
  <track kind=subtitles src=brave.de.vtt srclang=de lang=de
label="Deutsch">
</video>
```

(Les [lang](#)attributs sur les deux derniers décrivent la langue de l' [label](#)attribut, pas la langue des sous-titres eux-mêmes. La langue des sous-titres est donnée par l' [srclang](#)attribut.)

#### 4.8.11 Éléments multimédias

[Les objets HTMLMediaElement](#) ( [audio](#)et [video](#), dans cette spécification) sont simplement appelés **éléments multimédias** .

✓ MDN

```
enum CanPlayTypeResult { "" /* empty string */, "maybe",  
"probably" };
```

```
typedef (MediaStream or MediaSource or Blob) MediaProvider;
```

```
[Exposed=Window]
```

```
interface HTMLMediaElement : HTMLElement {
```

```
    // error state
```

```
    readonly attribute MediaError? error;
```

```
    // network state
```

```
    [CEReactions] attribute USVString src;
```

```
    attribute MediaProvider? srcObject;
```

```
    readonly attribute USVString currentSrc;
```

```
    [CEReactions] attribute DOMString? crossOrigin;
```

```
    const unsigned short NETWORK\_EMPTY = 0;
```

```
    const unsigned short NETWORK\_IDLE = 1;
```

```
    const unsigned short NETWORK\_LOADING = 2;
```

```
const unsigned short NETWORK\_NO\_SOURCE = 3;
```

```
readonly attribute unsigned short networkState;
```

```
[CEReactions] attribute DOMString preload;
```

```
readonly attribute TimeRanges buffered;
```

```
undefined load();
```

```
CanPlayTypeResult canPlayType(DOMString type);
```

```
// ready state
```

```
const unsigned short HAVE\_NOTHING = 0;
```

```
const unsigned short HAVE\_METADATA = 1;
```

```
const unsigned short HAVE\_CURRENT\_DATA = 2;
```

```
const unsigned short HAVE\_FUTURE\_DATA = 3;
```

```
const unsigned short HAVE\_ENOUGH\_DATA = 4;
```

```
readonly attribute unsigned short readyState;
```

```
readonly attribute boolean seeking;
```

```
// playback state
```

```
attribute double currentTime;
```

```
undefined fastSeek(double time);
```

```
readonly attribute unrestricted double duration;
```

```
object getStartDate();
```

```
readonly attribute boolean paused;
```

```
attribute double defaultPlaybackRate;
```



```
attribute double playbackRate;
```

```
attribute boolean preservesPitch;
```

```
readonly attribute TimeRanges played;
```

```
readonly attribute TimeRanges seekable;
```

```
readonly attribute boolean ended;
```

```
[CEReactions] attribute boolean autoplay;
```

```
[CEReactions] attribute boolean loop;
```

```
Promise<undefined> play();
```

```
undefined pause();
```

```
// controls
```

```
[CEReactions] attribute boolean controls;
```

```
attribute double volume;
```

```
attribute boolean muted;
```

```
[CEReactions] attribute boolean defaultMuted;
```

```
// tracks
```

```
[SameObject] readonly attribute AudioTrackList audioTracks;
```

```
[SameObject] readonly attribute VideoTrackList videoTracks;
```

```
[SameObject] readonly attribute TextTrackList textTracks;
```

```
TextTrack addTextTrack(TextTrackKind kind, optional DOMString  
label = "", optional DOMString language = "");
```

```
};
```

## Les attributs d'élément

**média** , [src](#), [crossorigin](#), [preload](#), [autoplay](#), [loop](#), [muted](#) et [controls](#), s'appliquent à tous [les éléments média](#) . Ils sont définis dans cette section.

[Les éléments multimédias](#) sont utilisés pour présenter des données audio, ou des données vidéo et audio, à l'utilisateur. Ceci est appelé **données multimédias** dans cette section, puisque cette section s'applique également aux [éléments multimédias](#) pour l'audio ou pour la vidéo. Le terme **ressource multimédia** est utilisé pour désigner l'ensemble complet de données multimédia, par exemple le fichier vidéo complet ou le fichier audio complet.

Une [ressource multimédia a une origine](#) associée , qui est soit " none " , " multiple " , " rewritten " ou une [origine](#) . Il est initialement défini sur " none " .

Une [ressource multimédia](#) peut avoir plusieurs pistes audio et vidéo. Pour les besoins d'un [élément multimédia](#) , les données vidéo de la [ressource multimédia](#) sont uniquement celles de la piste actuellement sélectionnée (le cas échéant) comme indiqué par l' [videoTracks](#) attribut de l'élément lorsque la [boucle d'événement](#) a atteint pour la dernière fois [l'étape 1](#) , et les données audio de la [ressource multimédia](#) est le résultat du mixage de toutes les pistes actuellement activées (le cas échéant) données par l' [audioTracks](#) attribut de l'élément lorsque la [boucle d'événement](#) a atteint pour la dernière fois [l'étape 1](#) .

*Les éléments [audio](#) et [video](#) peuvent être utilisés à la fois pour l'audio et la vidéo. La principale différence entre les deux est simplement que l' [audio](#) élément n'a pas de zone de lecture pour le contenu visuel (tel que la vidéo ou les sous-titres), alors que l' [video](#) élément en a une.*

Chaque [élément multimédia](#) a une **source de tâche d'événement d'élément multimédia** unique .

Pour **mettre en file d'attente une tâche d'élément multimédia** avec un [élément d'élément multimédia](#) et une série d'étapes étapes , mettez [une tâche d'élément en file d'attente](#) sur la [source de tâche d'événement d'élément multimédia de l'élément multimédia](#) donné *élément* et *étapes* donnés .

### 4.8.11.1 Codes d'erreur

✓ MDN

**`media.error`**

✓

Renvoie un [MediaError](#) objet représentant l'état d'erreur actuel de l'élément.  
Renvoie null s'il n'y a pas d'erreur.

Tous [les éléments multimédias](#) ont un état d'erreur associé, qui enregistre la dernière erreur rencontrée par l'élément depuis que son [algorithme de sélection de ressources](#) a été invoqué pour la dernière fois. L' **error**attribut, à l'obtention, doit retourner l' [MediaError](#)objet créé pour cette dernière erreur, ou null s'il n'y a pas eu d'erreur.

```
[Exposed=Window]

interface MediaError {

    const unsigned short MEDIA\_ERR\_ABORTED = 1;

    const unsigned short MEDIA\_ERR\_NETWORK = 2;

    const unsigned short MEDIA\_ERR\_DECODE = 3;

    const unsigned short MEDIA\_ERR\_SRC\_NOT\_SUPPORTED = 4;

    readonly attribute unsigned short code;

    readonly attribute DOMString message;

};
```

**media.error.code**

✓

Renvoie le code d'erreur de l'erreur actuelle, dans la liste ci-dessous.

**media.error.message**

✓

Renvoie un message de diagnostic informatif spécifique sur la condition d'erreur rencontrée. Le message et le format de message ne sont généralement pas uniformes entre les différents agents utilisateurs. Si aucun message de ce type n'est disponible, la chaîne vide est renvoyée.

Chaque [MediaError](#)objet a un **message** , qui est une chaîne, et un **code** , qui est l'un des suivants :

**[MEDIA\\_ERR\\_ABORTED](#)(valeur numérique 1)**

Le processus de récupération de la [ressource multimédia](#) a été interrompu par l'agent utilisateur à la demande de l'utilisateur.

**[MEDIA\\_ERR\\_NETWORK](#)(valeur numérique 2)**

Une erreur réseau de quelque nature que ce soit a amené l'agent utilisateur à arrêter de récupérer la [ressource multimédia](#) , après que la ressource a été établie comme étant utilisable.

**MEDIA\_ERR\_DECODE(valeur numérique 3)**

Une erreur quelconque s'est produite lors du décodage de la [ressource multimédia](#) , après que la ressource a été établie pour être utilisable.

**MEDIA\_ERR\_SRC\_NOT\_SUPPORTED(valeur numérique 4)**

La [ressource média](#) indiquée par l' [src](#) attribut ou [l'objet fournisseur de média attribué](#) n'était pas appropriée.

Pour **créer un** `MediaError` , étant donné un code d'erreur qui est l'une des valeurs ci-dessus, retournez un nouvel `MediaError` objet dont [le code](#) est le code d'erreur donné et dont [le message](#) est une chaîne contenant tous les détails que l'agent utilisateur est capable de fournir sur la cause de la condition d'erreur , ou la chaîne vide si l'agent utilisateur n'est pas en mesure de fournir ces détails. Cette chaîne de message ne doit pas contenir uniquement les informations déjà disponibles via le code d'erreur fourni ; par exemple, il ne doit pas s'agir simplement d'une traduction du code dans un format de chaîne. Si aucune information supplémentaire n'est disponible au-delà de celles fournies par le code d'erreur, le [message](#) doit être défini sur la chaîne vide.

Les étapes [du code](#) getter consistent à retourner [ce](#) code .

Les [étapes message](#) du getter consistent à renvoyer [ce](#) message .

#### 4.8.11.2 Emplacement de la ressource média

L' [src](#) attribut content sur [les éléments média](#) donne l' [URL](#) de la ressource média (vidéo, audio) à afficher. L'attribut, s'il est présent, doit contenir une [URL non vide valide potentiellement entourée d'espaces](#) .

Si l' [itemprop](#) attribut est spécifié sur l' [élément média](#) , alors l' [src](#) attribut doit également être spécifié.

L' [crossorigin](#) attribut de contenu sur [les éléments multimédias](#) est un [attribut de paramètres CORS](#) .

Si un [élément média](#) est créé avec un [src](#) attribut, l'agent utilisateur doit [immédiatement](#) invoquer l' [algorithme de sélection des ressources](#) de l' [élément média](#) .

Si un [src](#) attribut d'un [élément multimédia](#) est défini ou modifié, l'agent utilisateur doit invoquer l' [algorithme de chargement de l'élément](#) multimédia de l' élément

multimédia . ( *La suppression de l' attribut ne le fait pas, même s'il y a des éléments présents.*)[srcsource](#)



L' **src**attribut IDL sur [les éléments média](#) doit [réfléter](#) l'attribut content du même nom.



L' **crossOrigin**attribut IDL doit [réfléter](#) l' [crossorigin](#)attribut content, [limité aux seules valeurs connues](#) .

Un **objet fournisseur de médias** est un objet qui peut représenter une [ressource média](#) , distincte d' une [URL](#) . [MediaStream](#)les objets, [MediaSource](#)les objets et [Blob](#)les objets sont tous [des objets de fournisseur de médias](#) .

Chaque [élément multimédia](#) peut être **associé à un objet fournisseur multimédia** , qui est un [objet fournisseur multimédia](#) . Lorsqu'un [élément média](#) est créé, il n'a pas [d'objet de fournisseur de média assigné](#) .

```
media.srcObject [ = source ]
```



Permet à l' [élément média](#) d'être affecté à un [objet fournisseur de média](#) .

```
media.currentSrc
```



Renvoie l' [URL](#) de la [ressource multimédia](#) actuelle , le cas échéant.

Renvoie la chaîne vide lorsqu'il n'y a pas [de ressource multimédia](#) ou qu'elle n'a pas d' [URL](#) .

L' **currentSrc**attribut IDL doit initialement être défini sur la chaîne vide. Sa valeur est modifiée par l' [algorithme de sélection des ressources](#) défini ci-dessous.

L' **srcObject**attribut IDL, lors de l'obtention, doit renvoyer l' [objet fournisseur de médias attribué à](#) l'élément , le cas échéant, ou null dans le cas contraire. Lors de la définition, il doit définir l' [objet fournisseur de médias attribué à](#) l'élément sur la nouvelle valeur, puis invoquer l' [algorithme de chargement de l'élément multimédia](#) de l'élément .

*Il existe trois manières de spécifier une [ressource multimédia](#) : l' [srcObject](#)attribut IDL, l' **src**attribut de contenu et **source**les éléments. L'attribut IDL est prioritaire, suivi de l'attribut content, suivi des éléments.*

#### 4.8.11.3 Type MIME

Une [ressource multimédia](#) peut être décrite en termes de son *type*, spécifiquement un [type MIME](#), dans certains cas avec un `codecs` paramètre. (Le fait que le `codecs` paramètre soit autorisé ou non dépend du type MIME.) [\[RFC6381\]](#)

Les types sont généralement des descriptions quelque peu incomplètes ; par exemple "video/mpeg" ne dit rien sauf quel est le type de conteneur, et même un type comme "video/mp4; codecs="avc1.42E01E, mp4a.40.2"" n'inclut pas d'informations comme le débit réel (uniquement le débit maximum). Ainsi, étant donné un type, un agent utilisateur ne peut souvent savoir que s'il *peut* être capable de lire des médias de ce type (avec différents niveaux de confiance), ou s'il *ne peut* certainement pas lire des médias de ce type.

**Un type que l'agent utilisateur sait qu'il ne peut pas restituer** est celui qui décrit une ressource que l'agent utilisateur ne prend absolument pas en charge, par exemple parce qu'il ne reconnaît pas le type de conteneur ou qu'il ne prend pas en charge les codecs répertoriés.

Le [type MIME](#) "[application/octet-stream](#)" sans paramètres n'est jamais [un type dont l'agent utilisateur sait qu'il ne peut pas rendre](#). Les agents utilisateurs doivent traiter ce type comme l'équivalent de l'absence de toute [métadonnée Content-Type](#) explicite lorsqu'il est utilisé pour étiqueter une [ressource multimédia](#) potentielle.

*Seul le [type MIME](#) "[application/octet-stream](#)" sans paramètres a une casse spéciale ici ; si un paramètre apparaît avec lui, il sera traité comme n'importe quel autre [type MIME](#). Il s'agit d'un écart par rapport à la règle selon laquelle les paramètres de [type MIME](#) inconnus doivent être ignorés.*

`media.canPlayType (type)`

✓

*Renvoie la chaîne vide (une réponse négative), « peut-être » ou « probablement » en fonction de la confiance de l'agent utilisateur dans sa capacité à lire les ressources multimédias du type donné.*

La méthode doit renvoyer **la chaîne vide** si *type* est [un type que l'agent utilisateur sait qu'il ne peut pas restituer](#) ou est le type " " ; il doit retourner " " si l'agent utilisateur est sûr que le type représente une [ressource média](#) qu'il peut restituer s'il est utilisé avec cet élément ou ; et il doit retourner " " sinon. Les implémenteurs sont encouragés à retourner " " sauf si le type peut être établi avec certitude comme étant pris en charge ou non. Généralement, un agent utilisateur ne devrait jamais renvoyer " " pour un type qui autorise le paramètre si ce paramètre n'est pas présent. `canPlayType (type) application/octet-streamprobablyaudiovideomaybemaybeprobablycodecs`

Ce script teste pour voir si l'agent utilisateur prend en charge un nouveau format (fictif) pour décider dynamiquement d'utiliser un [video](#) élément ou un plugin :

```
<section id="video">
  <p><a href="playing-cats.nfv">Download video</a></p>
```

```

</section>
<script>
  var videoSection = document.getElementById('video');
  var videoElement = document.createElement('video');
  var support = videoElement.canPlayType('video/x-new-fictional-format;codecs="kittens,bunnies"');
  if (support != "probably" && "New Fictional Video Plugin" in navigator.plugins) {
    // not confident of browser support
    // but we have a plugin
    // so use plugin instead
    videoElement = document.createElement("embed");
  } else if (support == "") {
    // no support from browser and no plugin
    // do nothing
    videoElement = null;
  }
  if (videoElement) {
    while (videoSection.hasChildNodes())
      videoSection.removeChild(videoSection.firstChild);
    videoElement.setAttribute("src", "playing-cats.nfv");
    videoSection.appendChild(videoElement);
  }
</script>

```

*L' type attribut de l' source élément permet à l'agent utilisateur d'éviter de télécharger des ressources qui utilisent des formats qu'il ne peut pas restituer.*

#### 4.8.11.4 États du réseau

**media.networkState**



*Renvoie l'état actuel de l'activité réseau pour l'élément, à partir des codes de la liste ci-dessous.*

Lorsque les éléments multimédias interagissent avec le réseau, leur activité actuelle sur le réseau est représentée par l' **networkState** attribut. Lors de l'obtention, il doit renvoyer l'état actuel du réseau de l'élément, qui doit être l'une des valeurs suivantes :

**NETWORK\_EMPTY**(valeur numérique 0)

L'élément n'a pas encore été initialisé. Tous les attributs sont dans leur état initial.

#### **NETWORK\_IDLE(valeur numérique 1)**

L' [algorithme de sélection des ressources](#) de l'élément est actif et a sélectionné une [ressource](#) , mais il n'utilise pas réellement le réseau pour le moment.

#### **NETWORK\_LOADING(valeur numérique 2)**

L'agent utilisateur essaie activement de télécharger des données.

#### **NETWORK\_NO\_SOURCE(valeur numérique 3)**

L' [algorithme de sélection des ressources](#) de l'élément est actif, mais il n'a pas encore trouvé de [ressource](#) à utiliser.

L' [algorithme de sélection de ressources](#) défini ci-dessous décrit exactement quand l' `networkState` attribut change de valeur et quels événements se déclenchent pour indiquer des changements dans cet état.

### **4.8.11.5 Chargement de la ressource média**

`media.load()`



Provoque la réinitialisation de l'élément et le démarrage de la sélection et du chargement d'une nouvelle [ressource multimédia](#) à partir de zéro.

Tous [les éléments multimédias](#) ont un **indicateur can autoplay** , qui doit commencer à l'état true, et un **indicateur delaying-the-load-event** , qui doit commencer à l'état false. Tant que l' [indicateur delaying-the-load-event](#) est vrai, l'élément doit [retarder l'événement load](#) de son document.

Lorsque la `load()` méthode sur un [élément multimédia](#) est invoquée, l'agent utilisateur doit exécuter l' [algorithme de chargement de l'élément multimédia](#) .

L' **algorithme de chargement d'élément multimédia** comprend les étapes suivantes.

1. Abandonner toute instance déjà en cours d'exécution de l' [algorithme de sélection de ressources](#) pour cet élément.
2. Soit *les tâches en attente* une liste de toutes [les tâches](#) de la [source de tâche d'événement](#) de l'[élément multimédia](#) de l'élément multimédia dans l'une des [files d'attente de tâches](#) .
3. Pour chaque tâche dans *les tâches en attente* qui [résoudrait les promesses de lecture en attente](#) ou [rejetterait les promesses de lecture en attente](#) , résolvez



ou rejetez immédiatement ces promesses dans l'ordre dans lequel les tâches correspondantes ont été mises en file d'attente.

4. Supprimer chaque [tâche](#) des tâches en *attente* de sa [file d'attente de tâches](#)

*Fondamentalement, les événements et les rappels en attente sont ignorés et les promesses en cours à résoudre/rejeter sont résolues/rejetées immédiatement lorsque l'élément multimédia commence à charger une nouvelle ressource.*

5. Si l' [élément multimédia](#) est [networkState](#) défini sur [NETWORK\\_LOADING](#) ou [NETWORK\\_IDLE](#), [mettre en file d'attente une tâche d'élément multimédia](#) donnée à l' [élément multimédia](#) pour [déclencher un événement](#) nommé [abort](#) sur l' [élément multimédia](#) .
6. Si l' élément [mediaNetworkState](#) n'est pas défini sur [NETWORK\\_EMPTY](#), alors :
  1. [Mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [emptied](#) au niveau de l' [élément multimédia](#) .
  2. Si un processus de récupération est en cours pour l' [élément média](#) , l'agent utilisateur doit l'arrêter.
  3. Si l' [objet fournisseur de médias attribué](#) à l'[élément multimédia](#) est un objet, [détachez](#) -le. [MediaSource](#)
  4. [Oubliez les pistes spécifiques aux ressources multimédias de l'élément multimédia](#) .
  5. Si [readyState](#) n'est pas défini sur [HAVE NOTHING](#), réglez-le sur cet état.
  6. Si l' [paused](#) attribut est faux, alors :
    1. Définissez l' [paused](#) attribut sur true.
    2. [Prenez les promesses de jeu en attente](#) et [rejetez les promesses de jeu en attente](#) avec le résultat et un ["AbortError" DOMException](#) .
7. Si [seeking](#) est vrai, réglez-le sur faux.
8. Réglez la [position de lecture actuelle](#) sur 0.

Réglez la [position de lecture officielle](#) sur 0.

Si cela a changé la [position de lecture officielle](#) , mettez [en file d'attente une tâche d'élément multimédia](#) donnée à l' [élément](#)

[multimédia](#) pour [déclencher un événement](#) nommé [timeupdates](#) sur l' [élément multimédia](#) .

9. Définissez le [décalage de la chronologie](#) sur Not-a-Number (NaN).

10. Mettez à jour l' [duration](#) attribut sur Not-a-Number (NaN).

*L'agent utilisateur **ne** déclenchera pas d' [durationchange](#) événement pour ce changement particulier de durée.*

7. Définissez l' [playbackRate](#) attribut sur la valeur de l' [defaultPlaybackRate](#) attribut.

8. Définissez l' [error](#) attribut sur null et l' [indicateur can autoplay](#) sur true.

9. Invoque l' [algorithme de sélection des ressources](#) de l'[élément multimédia](#) .

*10. La lecture de toute [ressource multimédia](#) en cours de lecture pour cet élément s'arrête.*

L' **algorithme de sélection de ressources** pour un [élément multimédia](#) est le suivant. Cet algorithme est toujours invoqué dans le cadre d'une [tâche](#) , mais l'une des premières étapes de l'algorithme consiste à revenir en arrière et à poursuivre l'exécution des étapes restantes [en parallèle](#) . De plus, cet algorithme interagit étroitement avec le mécanisme [de boucle d'événements](#) ; en particulier, il a [des sections synchrones](#) (qui sont déclenchées dans le cadre de l' algorithme [de boucle d'événement](#) ). Les étapes de ces sections sont marquées d'un ⌚.

1. Définissez l' [networkState](#) attribut de l'élément sur la [NETWORK NO SOURCE](#) valeur.

2. Définissez l' [indicateur d'affiche d'exposition](#) de l'élément sur true.

3. Définissez le [drapeau delaying-the-load-event](#) de l'[élément média](#) sur true (cela [retarde l'événement load](#) ).

4. [Attendez un état stable](#) , permettant à la [tâche](#) qui a appelé cet algorithme de continuer. La [section synchrone](#) comprend toutes les étapes restantes de cet algorithme jusqu'à ce que l'algorithme indique que la [section synchrone](#) est terminée. (Les étapes des [sections synchrones](#) sont marquées d'un ⌚.)

5. ⌚ Si l' indicateur [de blocage sur l'analyseur](#) de l'[élément multimédia](#) est faux, [remplissez la liste des pistes de texte en attente](#) .

6. ⌚ Si l' [élément média](#) a un [objet fournisseur de média assigné](#) , alors laissez *mode* être *objet* .

⌚ Sinon, si l' [élément média](#) n'a pas [d'objet fournisseur de média assigné](#) mais a un [src](#) attribut, alors laissez *mode* être *attribut* .

⌚ Sinon, si l' [élément média](#) n'a pas d' [objet fournisseur de média assigné](#) et n'a pas d' [src](#)attribut, mais a un [source](#)enfant d'élément, alors laissez *mode* être *enfants* et laissez *candidat* être le premier [source](#)enfant de cet élément dans [l'ordre de l'arborescence](#) .

⌚ Sinon, l' [élément média](#) n'a pas d'[objet fournisseur de média assigné](#) et n'a ni [src](#)attribut ni [source](#)enfant d'élément :

1. ⌚ Réglez le [networkState](#)sur [NETWORK\\_EMPTY](#).
2. ⌚ Définissez l' [indicateur delaying-the-load-event](#) de l'élément sur false. Cela arrête [de retarder l'événement de chargement](#) .
3. Terminez la [section synchrone](#) et revenez.
7. ⌚ Réglez l' élément [multimédia](#)[networkState](#) sur [NETWORK\\_LOADING](#).
8. ⌚ [Mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [loadstart](#)à l' [élément multimédia](#) .
9. Exécutez les étapes appropriées dans la liste suivante :

#### **Si le mode est un objet**

1. ⌚ Définissez l' [currentSrc](#)attribut sur la chaîne vide.
2. Terminez la [section synchrone](#) en poursuivant les étapes restantes [en parallèle](#) .
3. Exécutez l' [algorithme d'extraction de ressources](#) avec l' [objet fournisseur de médias attribué](#) . Si cet algorithme revient sans abandonner *celui* -ci, alors le chargement a échoué.
4. *Échec avec le fournisseur de médias* : l'atteinte de cette étape indique que la ressource multimédia n'a pas pu être chargée. [Prenez les promesses de lecture en attente](#) et [mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour exécuter les [étapes d'échec de la source multimédia dédiée](#) avec le résultat.
5. Attendez que la [tâche](#) mise en file d'attente par l'étape précédente soit exécutée.
6. Retour. L'élément ne tentera pas de charger une autre ressource jusqu'à ce que cet algorithme soit à nouveau déclenché.

#### **Si le mode est un attribut**

7. ⌚ Si la [src](#) valeur de l'attribut est la chaîne vide, terminez la [section synchrone](#) et passez à l' étape *Échec avec l'attribut* ci-dessous.

8. ⌚ Soit *urlString* et *urlRecord* la [chaîne d'URL résultante](#) et l'[enregistrement d'URL résultant](#), respectivement, qui auraient résulté de l'[analyse](#) de l'[URL](#) spécifiée par la [src](#) valeur de l'attribut par rapport au [document de nœud](#) de l'[élément multimédia](#) lors de la dernière modification de l'attribut [src](#).
9. ⌚ Si *urlString* a été obtenu avec succès, définissez l'[currentSrc](#) attribut sur *urlString*.
10. Terminez la [section synchrone](#) en poursuivant les étapes restantes [en parallèle](#).
11. Si *urlRecord* a été obtenu avec succès, exécutez l'[algorithme de récupération de ressources](#) avec *urlRecord*. Si cet algorithme revient sans abandonner *celui-ci*, alors le chargement a échoué.
12. *Échec avec l'attribut* : Atteindre cette étape indique que la ressource multimédia n'a pas pu être chargée ou que l'[URL](#) donnée n'a pas pu être [analysée](#). [Prenez les promesses de lecture en attente](#) et [mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l'[élément multimédia](#) pour exécuter les [étapes d'échec de la source multimédia dédiée](#) avec le résultat.
13. Attendez que la [tâche](#) mise en file d'attente par l'étape précédente soit exécutée.
14. Retour. L'élément ne tentera pas de charger une autre ressource jusqu'à ce que cet algorithme soit à nouveau déclenché.

### **Sinon ( le mode est enfants )**

15. ⌚ Soit le *pointeur* une position définie par deux nœuds adjacents dans la liste des enfants de l'élément multimédia, en traitant le début de la liste (avant le premier enfant de la liste, le cas échéant) et la fin de la liste (après le dernier enfant de la liste, le cas échéant) en tant que nœuds à part entière. Un nœud est le nœud avant *le pointeur* et l'autre nœud est le nœud après *le pointeur*. Initialement, laissez *le pointeur* être la position entre le nœud *candidat* et le nœud suivant, s'il y en a, ou la fin de la liste, s'il s'agit du dernier nœud.

Au fur et à mesure que les nœuds sont [insérés](#) et [supprimés](#) dans l'[élément média](#), *le pointeur* doit être mis à jour comme suit :

**Si un nouveau nœud est [inséré](#) entre les deux nœuds qui définissent *le pointeur***

Soit *pointeur* le point entre le nœud avant *pointeur* et le nouveau nœud. En d'autres termes, les insertions au *pointeur* vont après *le pointeur*.

**Si le nœud avant *le pointeur* est supprimé**

Soit *pointeur* le point entre le nœud après *pointeur* et le nœud avant le nœud après *pointeur*. En d'autres termes, *le pointeur* ne se déplace pas par rapport aux nœuds restants.

### Si le nœud après le pointeur est supprimé

Soit *pointeur* le point entre le nœud avant *pointeur* et le nœud après le nœud avant *pointeur*. Tout comme dans le cas précédent, *le pointeur* ne se déplace pas par rapport aux nœuds restants.

Les autres modifications n'affectent pas *pointer*.

16. ⌚ *Traiter le candidat* : Si le *candidat* n'a pas d' srcattribut, ou si srcla valeur de son attribut est la chaîne vide, alors terminez la [section synchrone](#) et passez à l' étape ci-dessous *ayant échoué avec les éléments*.
17. ⌚ Soit *urlString* et *urlRecord* la [chaîne d'URL résultante](#) et l' [enregistrement d'URL résultant](#), respectivement, qui auraient résulté de l'[analyse](#) de l' [URL](#) spécifiée par la valeur de l'attribut du *candidat*src par rapport au [document de nœud](#) du *candidat* lors de la dernière modification de l'attribut.src
18. ⌚ Si *urlString* n'a pas été obtenu avec succès, terminez la [section synchrone](#) et passez à l' étape ci-dessous *ayant échoué avec les éléments*.
19. ⌚ Si le *candidat* a un typeattribut dont la valeur, lorsqu'il est analysé comme un [type MIME](#) (y compris tous les codecs décrits par le `codecs`paramètre, pour les types qui définissent ce paramètre), représente [un type que l'agent utilisateur sait qu'il ne peut pas restituer](#), puis terminez la [section synchrone](#), et passez à l' étape *échouée avec les éléments* ci-dessous.
20. ⌚ Définissez l' currentSrcattribut sur *urlString*.
21. Terminez la [section synchrone](#) en poursuivant les étapes restantes [en parallèle](#).
22. Exécutez l' [algorithme de récupération de ressources](#) avec *urlRecord*. Si cet algorithme revient sans abandonner *celui* -ci, alors le chargement a échoué.
23. *Échec avec les éléments* : [mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé errorau *candidat*.
24. [Attendez un état stable](#). La [section synchrone](#) comprend toutes les étapes restantes de cet algorithme jusqu'à ce que l'algorithme indique que la [section synchrone](#) est terminée. (Les étapes des [sections synchrones](#) sont marquées d'un ⌚.)
25. ⌚ [Oubliez les pistes spécifiques aux ressources multimédias de l'élément multimédia](#).

26. ⌚ Rechercher le candidat suivant : Soit candidat nul.
27. ⌚ Boucle de recherche : si le nœud après le pointeur est la fin de la liste, passez à l'étape d'attente ci-dessous.
28. ⌚ Si le nœud après le pointeur est un sourceélément, soit candidat cet élément.
29. ⌚ Avancez le pointeur de sorte que le nœud avant le pointeur soit maintenant le nœud qui était après le pointeur, et le nœud après le pointeur est le nœud après le nœud qui était après le pointeur, le cas échéant.
30. ⌚ Si le candidat est nul, revenez à l'étape de la boucle de recherche. Sinon, revenez à l'étape candidate du processus.
31. ⌚ Attente : Définissez l'networkStateattribut de l'élément sur la NETWORK NO SOURCEvaleur.
32. ⌚ Définissez le drapeau d'affichage de l'élément sur true.
33. ⌚ Mettre en file d'attente une tâche d'élément multimédia en fonction de l'élément multimédia pour définir l'indicateur d'événement de retardement du chargement sur faux. Cela arrête de retarder l'événement de chargement.
34. Terminez la section synchrone en poursuivant les étapes restantes en parallèle.
35. Attendez que le nœud après le pointeur soit un nœud autre que la fin de la liste. (Cette étape peut attendre éternellement.)
36. Attendez un état stable. La section synchrone comprend toutes les étapes restantes de cet algorithme jusqu'à ce que l'algorithme indique que la section synchrone est terminée. (Les étapes des sections synchrones sont marquées d'un ⌚.)
37. ⌚ Remettez l'indicateur delaying-the-load-event de l'élément à true (cela retarde à nouveau l'événement load, au cas où il n'aurait pas encore été déclenché).
38. ⌚ Réglez le networkStatedos sur NETWORK LOADING.
39. ⌚ Revenez à l'étape de recherche du prochain candidat ci-dessus.

Les **étapes d'échec de la source multimédia dédiée** avec une liste de promesses promesses sont les étapes suivantes :

40. Définissez l'errorattribut sur le résultat de la création d'unMediaError with MEDIA ERR SRC NOT SUPPORTED.

41. Oubliez les pistes spécifiques aux ressources multimédias de l'élément multimédia .
42. Définissez l' networkState attribut de l'élément sur la NETWORK NO SOURCE valeur.
43. Définissez l' indicateur d'affiche d'exposition de l'élément sur true.
44. Déclenchez un événement nommé error au niveau de l' élément média .
45. Rejeter les promesses de jeu en attente avec des promesses et un " NotSupportedError" DOMException .
46. Définissez l' indicateur delaying-the-load-event de l'élément sur false. Cela arrête de retarder l'événement de chargement .

Pour **vérifier une réponse multimédia** étant donné une réponse response , une ressource de ressource multimédia et " " ou un (nombre, nombre ou " ") tuple *byteRange* : `entire resourceuntil end`

1. Si la réponse est une erreur réseau , renvoie false.
2. Si *byteRange* est " `entire resource`", alors retourne true.
3. Soit *internalResponse* la réponse non sécurisée de la réponse .
4. Si le statut de *internalResponse* est 200, alors retourne true.
5. Si le statut de *internalResponse* n'est pas 206, renvoie false.
6. Si le résultat de l'extraction des valeurs de la plage de contenu à partir de *internalResponse* est un échec, renvoyez false.

*Notez que les valeurs extraites ne sont pas utilisées, et en particulier ne sont pas comparées à *byteRange* . Cette étape sert donc de validation syntaxique de l' Content-Range en-tête `` , mais si les Content-Range valeurs `` de la réponse ne correspondent pas aux Range valeurs `` de la requête, cela n'est pas considéré comme un échec.*

7. Soit *origin* être " `rewritten`" si l'URL de *internalResponse* est nulle ; sinon l' origine de l' URL de *internalResponse* .
8. Soit *previousOrigin* l' origine de la ressource .
9. Si l'une des conditions suivantes est vraie :
  - *previousOrigin* est " `none`" ;
  - *origin* et *previousOrigin* sont " `rewritten`" ; ou



- *origin* et *previousOrigin* sont [origins](#) , et *origin* est [la même origine](#) que *previousOrigin*

puis définissez [l'origine](#) de la *ressource* sur *origine* .

Sinon, si *la réponse* est [CORS-cross-origin](#) , renvoie *false*.

Sinon, définissez [l'origine](#) de la *ressource* sur " ".*multiple*

*Cela garantit que les réponses opaques avec des en-têtes de plage ne divulguent pas d'informations en étant regroupées avec d'autres réponses d'origines différentes.*

10. Renvoie vrai.

L' **algorithme d'extraction de ressources** pour un [élément multimédia](#) et un [enregistrement d'URL](#) donné ou [un objet fournisseur multimédia](#) est le suivant :

1. Si l'algorithme a été appelé avec [un objet de fournisseur de médias](#) ou un [enregistrement d'URL](#) dont [l'entrée d'URL de blob](#) est une [entrée d'URL de blob](#) dont [l'objet](#) est un [objet de fournisseur de médias](#) , alors laissez *mode* être *local* . Sinon, laissez *mode* à *distance* .
2. Si *mode* est *remote* , alors laissez la *ressource multimédia actuelle* être la ressource donnée par l' [enregistrement d'URL](#) passé à cet algorithme ; sinon, laissez la *ressource média actuelle* être la ressource donnée par l' [objet fournisseur de média](#) . Dans tous les cas, la *ressource multimédia actuelle* est désormais la [ressource multimédia](#) de l'élément .
3. Supprimez toutes [les pistes de texte spécifiques aux ressources multimédias](#) de la [liste des pistes de texte en attente](#) de l' [élément multimédia](#) , le cas échéant.
4. Exécutez les étapes appropriées dans la liste suivante :

#### **Si le *mode* est à distance**

1. Si vous le souhaitez, exécutez les sous-étapes suivantes. C'est le comportement attendu si l'agent utilisateur a l'intention de ne pas essayer de récupérer la ressource jusqu'à ce que l'utilisateur la demande explicitement (par exemple comme moyen d'implémenter le mot-clé [preload](#) de l'attribut [none](#)).
1. Réglez le [networkState](#) sur [NETWORK\\_IDLE](#).
2. [Mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [suspend](#) au niveau de l'élément.
3. [Mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour définir l' [indicateur](#)



[d'événement de retardement du chargement](#) sur faux. Cela arrête [de retarder l'événement de chargement](#) .

4. Attendez que la tâche soit exécutée.
  5. Attendre un événement [défini par l'implémentation](#) (par exemple, l'utilisateur demandant que l'élément multimédia commence la lecture).
  6. [Remettez l'indicateur delaying-the-load-event](#) de l'élément à true (cela [retarde à nouveau l'événement load](#) , au cas où il n'aurait pas encore été déclenché).
  7. Réglez le [networkState](#) sur [NETWORK\\_LOADING](#).
2. Soit *destination* " audio" si l' [élément média](#) est un [audio](#)élément, ou " video" sinon.
  3. Supposons que *request* soit le résultat de [la création d'une requête CORS potentielle](#) en fonction [de l'enregistrement d'URL](#) de *la ressource multimédia actuelle* , de *la destination* et de l'état actuel de l'attribut de contenu de [l'élément multimédia](#) .[crossorigin](#)
  4. Définissez le [client de](#) *la requête* sur l' [objet de paramètres pertinent](#) du [document de nœud](#) de [l'élément multimédia](#) .
  5. Définissez [le type d'initiateur](#) de *la requête* sur *destination* .
  6. Soit *byteRange* , qui est " `entire resource`" ou un `until end` tuple (nombre, nombre ou " "), soit la plage d'octets requise pour satisfaire les données manquantes dans [media data](#) . Cette valeur est [définie par l'implémentation](#) et peut dépendre du codec, des conditions du réseau ou d'autres heuristiques. L'agent utilisateur peut décider d'extraire la ressource dans son intégralité, auquel cas *byteRange* serait " `entire resource`", d'extraire d'un décalage d'octet jusqu'à la fin, auquel cas *byteRange* serait (nombre, " `until end`"), ou d'extraire une plage entre deux décalages d'octets, im auquel cas *byteRange* serait un tuple (nombre, nombre) représentant les deux décalages.
  7. Si *byteRange* n'est pas " `entire resource`", alors :
    1. Si *byteRange* [ 1] est " `until end`" alors [ajoutez un en-tête de plage](#) pour *demandeur byteRange* [0].
    2. Sinon, [ajoutez un en-tête de plage](#) pour *demandeur les byteRange* [0] et *byteRange* [ 1 ] .
  8. [Récupérez](#) *la requête* , avec [processResponse](#) défini sur les étapes suivantes en fonction de *la réponse* :
    1. Soit *global* l' [objet global pertinent](#) du [document de nœud](#) de [l'élément média](#) .

2. Soit *updateMedia* pour mettre [en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour exécuter les premières étapes appropriées à partir de la [liste des étapes de traitement des données multimédias](#) ci-dessous. (Une nouvelle tâche est utilisée pour cela afin que le travail décrit ci-dessous se produise par rapport à la [source de tâche d'événement d'élément multimédia](#) appropriée plutôt que d'utiliser la [source de tâche réseau](#) .)
3. Soit *processEndOfMedia* l'étape suivante : si le processus de récupération s'est terminé sans erreur, y compris le décodage des données multimédias, et si toutes les données sont disponibles pour l'agent utilisateur sans accès au réseau, alors, l'agent utilisateur doit passer à l'étape *finale* dessous. Cela peut ne jamais se produire, par exemple lors de la diffusion d'une ressource infinie telle qu'une radio Web, ou si la ressource est plus longue que la capacité de l'agent utilisateur à mettre en cache les données.
4. Si le résultat de [la vérification de la réponse](#) compte tenu de la *ressource multimédia actuelle* et de la *plage d'octets* est faux, abandonnez ces étapes.
5. Sinon, [lisez de manière incrémentielle le corps](#) de la *réponse* avec *updateMedia* , *processEndOfMedia* , un algorithme vide et *global* .
6. Mettez à jour les [données multimédias](#) avec le contenu de la *réponse non sécurisée de réponse* obtenue de cette manière. *la réponse* peut être [CORS-same-origin](#) ou [CORS-cross-origin](#) ; cela affecte si les sous-titres référencés dans les [données multimédias](#) sont exposés dans l'API et, pour [video](#) les éléments, si un [canvas](#) est entaché lorsque la vidéo est dessinée dessus.

Le **délai d'attente de blocage de l'élément multimédia** est une durée [définie par l'implémentation , qui doit être d'environ trois secondes](#). Lorsqu'un [élément multimédia](#) qui tente activement d'obtenir [des données multimédias](#) n'a pas reçu de données pendant une durée égale au [délai d'attente de l'élément multimédia](#) , l'agent utilisateur doit [mettre en file d'attente une tâche d'élément multimédia](#) donnée à l' [élément multimédia](#) pour [déclencher un événement](#) nommé [stalled](#) à l'élément .

Les agents utilisateurs peuvent permettre aux utilisateurs de bloquer ou de ralentir sélectivement les téléchargements [de données multimédias](#) . Lorsque le téléchargement d'un [élément multimédia](#) a été complètement bloqué, l'agent utilisateur doit agir comme s'il était bloqué (au lieu d'agir comme si la connexion était fermée). Le taux de téléchargement peut également être limité automatiquement par l'agent

utilisateur, par exemple pour équilibrer le téléchargement avec d'autres connexions partageant la même bande passante.

Les agents utilisateurs peuvent décider de ne pas télécharger plus de contenu à tout moment, par exemple après avoir mis en mémoire tampon cinq minutes d'une ressource multimédia d'une heure, en attendant que l'utilisateur décide de lire ou non la ressource, en attendant l'entrée de l'utilisateur dans une ressource interactive, ou lorsque l'utilisateur quitte la page. Lorsque le téléchargement d'un [élément multimédia](#) a été suspendu, l'agent utilisateur doit [mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour définir le [networkState](#) à [NETWORK\\_IDLE](#) et [déclencher un événement](#) nommé [suspend](#) sur l'élément. Si et lorsque le téléchargement de la ressource reprend, l'agent utilisateur doit [mettre en file d'attente une tâche d'élément multimédia](#) compte tenu de l' [élément multimédia](#) pour régler le [networkState](#) à [NETWORK\\_LOADING](#). Entre la mise en file d'attente de ces tâches, le chargement est suspendu ( [progress](#) les événements ne se déclenchent donc pas, comme décrit ci-dessus).

*L' [preload](#) attribut fournit une indication sur la quantité de mémoire tampon que l'auteur juge souhaitable, même en l'absence de l' [autoplay](#) attribut.*

Lorsqu'un agent utilisateur décide de suspendre complètement un téléchargement, par exemple, s'il attend que l'utilisateur démarre la lecture avant de télécharger tout autre contenu, l'agent utilisateur doit mettre en file d'attente une tâche d'élément multimédia [en](#) fonction de l' [élément multimédia](#) pour définir le [délai de chargement de l'élément. -drapeau d'événement](#) à faux. Cela arrête [de retarder l'événement de chargement](#) .

Bien que les étapes ci-dessus donnent un algorithme pour émettre des requêtes, l'agent utilisateur peut utiliser d'autres moyens que ceux-là, en particulier face à des conditions d'erreur. Par exemple, l'agent utilisateur peut se reconnecter au serveur ou passer à un protocole de diffusion en continu. L'agent utilisateur ne doit considérer la ressource comme erronée et passer aux branches d'erreur des étapes ci-dessus que si l'agent utilisateur a renoncé à chercher la ressource.

Pour déterminer le format de la [ressource multimédia](#) , l' agent utilisateur doit utiliser les [règles de reniflage audio et vidéo spécifiquement](#) .

Tant que le chargement n'est pas suspendu (voir ci-dessous), toutes les 350 ms ( $\pm$  200 ms) ou pour chaque octet reçu, selon la fréquence la *moins* fréquente, [mettez en file d'attente une tâche d'élément multimédia](#) donnée à l' [élément multimédia](#) pour [déclencher un événement](#) nommé [progress](#) à l'élément.

Bien que l'agent utilisateur puisse toujours avoir besoin d'un accès au réseau pour obtenir des parties de la [ressource multimédia](#) , l'agent utilisateur doit rester à cette étape.

Par exemple, si l'agent utilisateur a supprimé la première moitié d'une vidéo, l'agent utilisateur restera à cette étape même une fois la [lecture terminée](#) , car il est toujours possible que l'utilisateur revienne au début. En fait, dans cette situation, une fois [la lecture terminée](#) , l'agent utilisateur finira par déclencher un [suspend](#) événement, comme décrit précédemment.

### **Sinon ( le mode est local )**

La ressource décrite par la *ressource multimédia actuelle* , le cas échéant, contient les [données multimédias](#) . C'est [CORS-même-origine](#) .

Si la *ressource multimédia actuelle* est un flux de données brutes (par exemple, à partir d'un [File](#) objet), alors pour déterminer le format de la [ressource multimédia](#) , l'agent utilisateur doit utiliser les [règles de reniflage audio et vidéo spécifiquement](#) . Sinon, si le flux de données est pré-décodé, alors le format est le format donné par la spécification pertinente.

Chaque fois que de nouvelles données pour la *ressource multimédia actuelle* deviennent disponibles, [mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour exécuter les premières étapes appropriées à partir de la [liste des étapes de traitement des données multimédia](#) ci-dessous.

Lorsque la *ressource média courante* est définitivement épuisée (par exemple tous les octets de a [Blob](#) ont été traités), s'il n'y a pas eu d'erreurs de décodage, alors l'agent utilisateur doit passer à l' *étape finale* ci-dessous. Cela peut ne jamais arriver, par exemple si la *ressource multimédia actuelle* est un fichier [MediaStream](#).

La **liste des étapes de traitement des données multimédias** est la suivante :

**Si les [données multimédias](#) ne peuvent pas être récupérées du tout, en raison d'erreurs de réseau, l'agent utilisateur renonce à essayer de récupérer la ressource**

**Si les [données multimédias](#) peuvent être récupérées mais que l'inspection révèle qu'elles sont dans un format non pris en charge ou qu'elles ne peuvent pas être restituées du tout**

Erreurs DNS, erreurs HTTP 4xx et 5xx (et équivalents dans d'autres protocoles) et autres erreurs réseau fatales qui se produisent avant que l'agent utilisateur n'ait établi si la *ressource multimédia actuelle* est utilisable, ainsi que le fichier utilisant un format de conteneur non pris en charge, ou utilisant codecs non pris en charge pour toutes les données, doit obliger l'agent utilisateur à exécuter les étapes suivantes :

9. L'agent utilisateur doit annuler le processus de récupération.

10. Abandonnez ce sous-algorithme et revenez à l' [algorithme de sélection des ressources](#) .

**S'il s'avère que la [ressource multimédia](#) possède une piste audio**

11. Créez un [AudioTrack](#) objet pour représenter la piste audio.
12. Mettez à jour l' objet de l'attribut de [l'élément multimédia](#) avec le nouvel objet. [audioTracksAudioTrackListAudioTrack](#)
13. Soit *enable* inconnu . \_
14. Si la [ressource multimédia](#) ou l' [URL](#) de la *ressource multimédia actuelle* indique un ensemble particulier de pistes audio à activer, ou si l'agent utilisateur dispose d'informations qui faciliteraient la sélection de pistes audio spécifiques pour améliorer l'expérience de l'utilisateur, alors : si cette piste audio track est l'un de ceux à activer, puis définissez *enable* sur *true* , sinon, définissez *enable* sur *false* .

Cela pourrait être déclenché par [la syntaxe des fragments de média](#) , mais cela pourrait également être déclenché par exemple par l'agent utilisateur sélectionnant une piste audio de son surround 5.1 sur une piste audio stéréo.

15. Si *enable* est encore *unknown* , alors, si l' [élément multimédia](#) n'a pas encore de piste audio [activée](#) , alors réglez *enable* sur *true* , sinon, réglez *enable* sur *false* .
16. Si *enable* est *true* , alors activez cette piste audio, sinon, n'activez pas cette piste audio.
17. [Lancez un événement](#) nommé [addtrack](#) sur cet [AudioTrackList](#) objet, en utilisant [TrackEvent](#), avec l' [track](#) attribut initialisé sur le nouvel [AudioTrack](#) objet.

**S'il s'avère que la [ressource multimédia](#) possède une piste vidéo**

18. Créez un [VideoTrack](#) objet pour représenter la piste vidéo.
19. Mettez à jour l' objet de l'attribut de [l'élément multimédia](#) avec le nouvel objet. [videoTracksVideoTrackListVideoTrack](#)
20. Soit *enable* inconnu . \_
21. Si la [ressource multimédia](#) ou l' [URL](#) de la *ressource multimédia actuelle* indique un ensemble particulier de pistes vidéo à activer, ou si l'agent utilisateur dispose d'informations qui faciliteraient la sélection de pistes vidéo spécifiques pour améliorer l'expérience de l'utilisateur, alors : si cette vidéo track est la première piste vidéo de ce type, puis définissez *enable* sur *true* , sinon, définissez *enable* sur *false* .

Cela pourrait à nouveau être déclenché par [la syntaxe des fragments de média](#) .

22. Si *enable* est encore *inconnu* , alors, si l' [élément multimédia](#) n'a pas encore de piste vidéo [sélectionnée](#) , alors réglez *enable* sur *true* , sinon, réglez *enable* sur *false* .
23. Si *enable* est *true* , sélectionnez cette piste et désélectionnez toutes les pistes vidéo précédemment sélectionnées, sinon ne sélectionnez pas cette piste vidéo. Si d'autres pistes ne sont pas sélectionnées, [un changement d'événement sera déclenché](#).
24. [Lancez un événement](#) nommé `addtrack` sur cet `VideoTrackList` objet, en utilisant `TrackEvent`, avec l' `track` attribut initialisé sur le nouvel `VideoTrack` objet.

Une fois que suffisamment de [données multimédias](#) ont été extraites pour déterminer la durée de la [ressource multimédia](#) , ses dimensions et d'autres métadonnées

Cela indique que la ressource est utilisable. L'agent utilisateur doit suivre ces sous-étapes :

25. [Établir la chronologie des médias](#) pour les besoins de la [position de lecture actuelle](#) et la [première position possible](#) , en fonction des [données multimédias](#) .
26. Mettez à jour le [décalage de la chronologie](#) à la date et à l'heure qui correspondent à l'heure zéro dans la [chronologie des médias](#) établie à l'étape précédente, le cas échéant. Si aucune heure et date explicites ne sont fournies par la [ressource multimédia](#) , le [décalage de la chronologie](#) doit être défini sur Not-a-Number (NaN).
27. Réglez la [position de lecture actuelle](#) et la [position de lecture officielle](#) sur la [première position possible](#) .
28. Mettez à jour l' `duration` attribut avec l'heure de la dernière image de la ressource, si elle est connue, sur la [chronologie des médias](#) établie ci-dessus. S'il n'est pas connu (par exemple, un flux qui est en principe infini), mettez à jour l' `duration` attribut avec la valeur positive Infinity.

*L'agent utilisateur mettra en file d'attente une tâche d'élément multimédia donnée à l' [élément multimédia](#) pour [déclencher un événement](#) nommé `durationchange` à l'élément à ce stade.*

29. Pour `video` les éléments, définissez les attributs `videoWidth` et `videoHeight`, et [mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé `resize` au niveau de l' [élément multimédia](#) .



*D'autres resize événements seront déclenchés si les dimensions changent par la suite.*

30. Définissez l' readyState attribut sur HAVE METADATA.

*Un loadedmetadata événement DOM sera déclenché dans le cadre de la définition de l' readyState attribut sur une nouvelle valeur.*

31. Soit *sauté* faux.

32. Si la position de début de lecture par défaut de l'élément multimédia est supérieure à zéro, recherchez cette heure et laissez *jumped* être vrai.

33. Laissez la position de début de lecture par défaut de l' élément multimédia à zéro.

34. Laissez la *position de lecture initiale* être zéro.

35. Si la ressource multimédia ou l' URL de la *ressource multimédia* actuelle indique une heure de début particulière, définissez la *position de lecture initiale* sur cette heure et, si *jumped* est toujours faux, recherchez cette heure.

Par exemple, avec les formats de média prenant en charge la syntaxe de fragment de média , le fragment peut être utilisé pour indiquer une position de départ.

36. S'il n'y a pas de piste audio activée , activez une piste audio. Cela provoquera change le déclenchement d'un événement .

37. S'il n'y a pas de piste vidéo sélectionnée , sélectionnez une piste vidéo. Cela provoquera change le déclenchement d'un événement .

Une fois que l' readyState attribut atteint HAVE CURRENT DATA, après le loadeddata déclenchement de l'événement , définissez l' indicateur delaying-the-load-event de l'élément sur false. Cela arrête de retarder l'événement de chargement .

*Un agent utilisateur qui tente de réduire l'utilisation du réseau tout en récupérant les métadonnées pour chaque ressource multimédia arrêterait également la mise en mémoire tampon à ce stade, en suivant les règles décrites précédemment , qui impliquent le networkState passage de l'attribut à la NETWORK\_IDLE valeur et suspend le déclenchement d'un événement. L'agent utilisateur doit déterminer la durée de la ressource multimédia et passer par cette étape avant de jouer.*

Une fois que l'intégralité de la ressource multimédia a été récupérée (mais potentiellement avant qu'aucune d'entre elles n'ait été décodée)

Déclenchez un événement nommé progress au niveau de l' élément média .

Définissez le networkState sur NETWORK IDLE et déclenchez un événement nommé suspend à l'élément média.

Si jamais l'agent utilisateur supprime des données multimédias et doit ensuite reprendre l'activité réseau pour les obtenir à nouveau, il doit alors mettre en file d'attente une tâche d'élément multimédia en fonction de l'élément multimédia à définir networkState sur NETWORK LOADING.

*Si l'agent utilisateur peut garder la ressource média chargée, alors l'algorithme continuera à sa dernière étape ci-dessous, qui interrompt l'algorithme.*

**Si la connexion est interrompue après la réception de certaines données multimédias, ce qui oblige l'agent utilisateur à abandonner la tentative de récupération de la ressource**

Les erreurs réseau fatales qui se produisent après que l'agent utilisateur a établi si la ressource média actuelle est utilisable (c'est-à-dire une fois que l'attribut de l'élément média readyState n'est plus HAVE NOTHING) doivent obliger l'agent utilisateur à exécuter les étapes suivantes :

38. L'agent utilisateur doit annuler le processus de récupération.

39. Définissez l'error attribut sur le résultat de la création d'un MediaError with MEDIA\_ERR\_NETWORK.

40. Définissez l'networkState attribut de l'élément sur la NETWORK\_IDLE valeur.

41. Définissez l'indicateur delaying-the-load-event de l'élément sur false. Cela arrête de retarder l'événement de chargement.

42. Déclenchez un événement nommé error au niveau de l'élément média.

43. Abandonner l'algorithme global de sélection des ressources.

**Si les données multimédias sont corrompues**

Les erreurs fatales dans le décodage des données multimédias qui se produisent après que l'agent utilisateur a établi si la ressource multimédia actuelle est utilisable (c'est-à-dire une fois que l'attribut de l'élément multimédia readyState n'est plus HAVE NOTHING) doivent obliger l'agent utilisateur à exécuter les étapes suivantes :

44. L'agent utilisateur doit annuler le processus de récupération.

45. Définissez l'error attribut sur le résultat de la création d'un MediaError with MEDIA\_ERR\_DECODE.

46. Définissez l'networkState attribut de l'élément sur la NETWORK\_IDLE valeur.



47. [Définissez l'indicateur delaying-the-load-event](#) de l'élément sur false. Cela arrête [de retarder l'événement de chargement](#).
48. [Déclenchez un événement](#) nommé [error](#) au niveau de l'[élément média](#).
49. Abandonner l' [algorithme global de sélection des ressources](#).

**Si le processus de récupération [des données multimédias](#) est interrompu par l'utilisateur**

Le processus de récupération est interrompu par l'utilisateur, par exemple parce que l'utilisateur a appuyé sur un bouton "stop", l'agent utilisateur doit exécuter les étapes suivantes. Ces étapes ne sont pas suivies si la [load\(\)](#) méthode elle-même est appelée pendant l'exécution de ces étapes, car les étapes ci-dessus gèrent ce type particulier d'abandon.

50. L'agent utilisateur doit annuler le processus de récupération.
51. Définissez l' [error](#) attribut sur le résultat de [la création d'un `MediaError` with `MEDIA\_ERR\_ABORTED`](#).
52. [Déclenchez un événement](#) nommé [abort](#) au niveau de l' [élément média](#).
53. Si l' attribut de l' [élément média](#) [readyState](#) a une valeur égale à [HAVE\\_NOTHING](#), définissez l'attribut de l'élément [networkState](#) sur la [NETWORK\\_EMPTY](#) valeur, définissez l' [indicateur d'affichage](#) de l'élément sur true et [déclenchez un événement](#) nommé [emptied](#) sur l'élément.  
  
Sinon, définissez l' [networkState](#) attribut de l'élément sur la [NETWORK\\_IDLE](#) valeur.
54. [Définissez l'indicateur delaying-the-load-event](#) de l'élément sur false. Cela arrête [de retarder l'événement de chargement](#).
55. Abandonner l' [algorithme global de sélection des ressources](#).

**Si les [données multimédias](#) peuvent être récupérées mais comportent des erreurs non fatales ou utilisent, en partie, des codecs non pris en charge, empêchant l'agent utilisateur de restituer le contenu complètement correctement mais n'empêchant pas complètement la lecture**

Le serveur qui renvoie des données partiellement utilisables mais qui ne peuvent pas être restituées de manière optimale doit obliger l'agent utilisateur à restituer uniquement les bits qu'il peut gérer et à ignorer le reste.

**Si la [ressource multimédia](#) est trouvée pour déclarer une [piste de texte spécifique à la ressource multimédia](#) que l'agent utilisateur prend en charge**

Si les [données multimédias](#) sont [CORS-same-origin](#) , exécutez les [étapes pour exposer une piste de texte spécifique à la ressource multimédia](#) avec les données pertinentes.

*Les vidéos cross-origin n'exposent pas leurs sous-titres, car cela permettrait des attaques telles que des sites hostiles lisant les sous-titres de vidéos confidentielles sur l'intranet d'un utilisateur.*

5. *Dernière étape* : si jamais l'agent utilisateur atteint cette étape (ce qui ne peut se produire que si la totalité de la ressource est chargée et maintenue disponible) : abandonnez l' [algorithme de sélection de ressource](#) global .

Lorsqu'un [élément média](#) doit **oublier les pistes spécifiques à la ressource média de l'élément média** , l'agent utilisateur doit supprimer de la [liste des pistes de texte](#) de l'[élément média](#) toutes les [pistes de texte spécifiques à la ressource média](#) , puis vider l' attribut de l' [élément média](#) objet, puis videz l' objet de l'attribut de [l'élément média](#) . Aucun événement (en particulier, aucun événement) n'est déclenché dans le cadre de cela ; les événements et , déclenchés par les algorithmes qui invoquent celui-ci, peuvent être utilisés à la place.[audioTracksAudioTrackListvideoTracksVideoTrackListremoveTrackererroremptied](#)

L' **preload** attribut est un [attribut énuméré](#) . Le tableau suivant répertorie les mots-clés et les états de l'attribut — les mots-clés de la colonne de gauche correspondent aux états de la cellule de la deuxième colonne sur la même ligne que le mot-clé. L'attribut peut être modifié même une fois que la [ressource multimédia](#) est mise en mémoire tampon ou lue ; les descriptions du tableau ci-dessous doivent être interprétées dans cet esprit.

Mot-clé	État	Brève description
<b>none</b>	<b>Aucun</b>	Indique à l'agent utilisateur que soit l'auteur ne s'attend pas à ce que l'utilisateur ait besoin de la ressource multimédia, soit que le serveur souhaite minimiser le trafic inutile. Cet état ne fournit pas d'indication sur l'agressivité avec laquelle télécharger réellement la ressource multimédia si la mise en mémoire tampon démarre de toute façon (par exemple, une fois que l'utilisateur appuie sur "jouer").
<b>metadata</b>	<b>Métadonnées</b>	Indique à l'agent utilisateur que l'auteur ne s'attend pas à ce que l'utilisateur ait besoin de la ressource multimédia, mais que la récupération des métadonnées de la ressource (dimensions, liste des pistes, durée, etc.), et peut-être même des premières images, est raisonnable. Si l'agent utilisateur ne récupère précisément pas plus que les métadonnées,

Mot-clé	État	Brève description
		l' <a href="#">élément média</a> se retrouvera avec son <a href="#">readyState</a> attribut défini sur <a href="#">HAVE METADATA</a> ; généralement cependant, certaines images seront également obtenues et ce sera probablement <a href="#">HAVE CURRENT DATA</a> ou <a href="#">HAVE FUTURE DATA</a> . Lorsque la ressource multimédia est en cours de lecture, indique à l'agent utilisateur que la bande passante doit être considérée comme rare, par exemple en suggérant de limiter le téléchargement afin que les données multimédias soient obtenues au rythme le plus lent possible tout en maintenant une lecture cohérente.
<a href="#">auto</a>	<b>Automatique</b>	Indique à l'agent utilisateur que l'agent utilisateur peut donner la priorité aux besoins de l'utilisateur sans risque pour le serveur, jusqu'à et y compris le téléchargement optimiste de la totalité de la ressource.

La chaîne vide est également un mot-clé valide et correspond à l' état [Automatique](#) . La [valeur manquante par défaut](#) et [la valeur invalide par défaut](#) de l'attribut sont [définies par l'implémentation](#) , bien que l' état [des métadonnées](#) soit suggéré comme compromis entre la réduction de la charge du serveur et l'offre d'une expérience utilisateur optimale.

*Les auteurs peuvent changer l'attribut de "[none](#)" ou "[metadata](#)" à "[auto](#)" dynamiquement une fois que l'utilisateur commence la lecture. Par exemple, sur une page avec de nombreuses vidéos, cela peut être utilisé pour indiquer que les nombreuses vidéos ne doivent pas être téléchargées à moins d'être demandées, mais qu'une fois qu'une est demandée, elle doit être téléchargée de manière agressive.*

L' [preload](#)attribut est destiné à fournir une indication à l'agent utilisateur sur ce que l'auteur pense conduira à la meilleure expérience utilisateur. L'attribut peut être complètement ignoré, par exemple sur la base de préférences explicites de l'utilisateur ou sur la base de la connectivité disponible.

L' [preload](#) attribut IDL doit [réfléter](#) l'attribut de contenu du même nom, [limité aux seules valeurs connues](#) .

*L' [autoplay](#)attribut peut remplacer l' [preload](#)attribut (puisque si le média est lu, il doit naturellement d'abord mettre en mémoire tampon, quelle que soit l'indication donnée par l' [preload](#)attribut). Cependant, inclure les deux n'est pas une erreur.*

---

**[media.buffered](#)**



Renvoie un [TimeRanges](#) objet qui représente les plages de la [ressource multimédia](#) que l'agent utilisateur a mises en mémoire tampon.

L' **buffered** attribut doit renvoyer un nouvel [objet normalisé TimeRanges](#) statique qui représente les plages de la [ressource média](#) , le cas échéant, que l'agent utilisateur a mises en mémoire tampon, au moment où l'attribut est évalué. Les agents utilisateurs doivent déterminer avec précision les plages disponibles, même pour les flux multimédias où cela ne peut être déterminé que par une inspection fastidieuse.

*Typiquement, ce sera une plage unique ancrée au point zéro, mais si, par exemple, l'agent utilisateur utilise des requêtes de plage HTTP en réponse à la recherche, alors il pourrait y avoir plusieurs plages.*

Les agents utilisateurs peuvent supprimer les données précédemment mises en mémoire tampon.

*Ainsi, une position temporelle incluse dans une plage d'objets renvoyés par l' **buffered** attribut à un moment donné peut finir par ne pas être incluse dans la ou les plages d'objets renvoyés par le même attribut à des instants ultérieurs.*

***Retourner un nouvel objet à chaque fois est un mauvais modèle pour les getters d'attributs et n'est inscrit ici que car il serait coûteux de le changer. Il ne doit pas être copié dans de nouvelles API.***

#### 4.8.11.6 Décalages dans la ressource média

**media.duration**



Renvoie la longueur de la [ressource média](#) , en secondes, en supposant que le début de la [ressource média](#) est à l'instant zéro.

Renvoie NaN si la durée n'est pas disponible.

Renvoie Infinity pour les flux illimités.

**media.currentTime** [ = value ]



Renvoie la [position de lecture officielle](#) , en secondes.

Peut être réglé, pour chercher à l'heure donnée.

Une [ressource multimédia](#) a une **chronologie multimédia** qui mappe les temps (en secondes) aux positions dans la [ressource multimédia](#) . L'origine d'une chronologie est sa première position définie. La durée d'une chronologie est sa dernière position définie.

**Établir la chronologie des médias** : si la [ressource multimédia](#) spécifie d'une manière ou d'une autre une chronologie explicite dont l'origine n'est pas négative (c'est-à-dire donne à chaque image un décalage temporel spécifique et donne à la première image un décalage nul ou positif), alors la chronologie des médias devrait être cette [chronologie](#) . (Le fait que la [ressource multimédia](#) puisse ou non spécifier une chronologie dépend du format [de la ressource multimédia](#) .) Si la [ressource multimédia](#) spécifie une heure *et une date* de début explicites , alors cette heure et cette date doivent être considérées comme le point zéro de la [chronologie multimédia](#) ; le [décalage de la chronologie](#) sera l'heure et la date, exposées à l'aide de la [getStartDate\(\)](#) méthode.

Si la [ressource multimédia](#) a une chronologie discontinue, l'agent utilisateur doit étendre la chronologie utilisée au début de la ressource sur l'ensemble de la ressource, de sorte que la [chronologie multimédia](#) de la [ressource multimédia](#) augmente de manière linéaire à partir de la [position la plus précoce possible](#) (comme défini ci-dessous) , même si les [données multimédias](#) sous-jacentes ont des codes temporels dans le désordre ou même qui se chevauchent.

Par exemple, si deux clips ont été concaténés dans un seul fichier vidéo, mais que le format vidéo expose les heures d'origine des deux clips, les données vidéo peuvent exposer une chronologie qui va, par exemple, 00:15..00:29 puis 00:05..00:38. Cependant, l'agent utilisateur n'exposerait pas ces heures ; il exposerait plutôt les heures comme 00:15..00:29 et 00:29..01:02, comme une seule vidéo.

Dans les rares cas où une [ressource multimédia](#) n'a pas de chronologie explicite, l'heure zéro sur la [chronologie du média](#) doit correspondre à la première image de la [ressource multimédia](#) . Dans le cas encore plus rare d'une [ressource multimédia](#) sans aucun timing explicite d'aucune sorte, pas même des durées de trame, l'agent utilisateur doit lui-même déterminer le temps pour chaque trame d'une manière [définie par l'implémentation](#) .

*Un exemple de format de fichier sans chronologie explicite mais avec des durées d'image explicites est le format GIF animé. Un exemple de format de fichier sans aucun timing explicite est le format JPEG-push ([multipart/x-mixed-replace](#) avec des images JPEG, souvent utilisé comme format pour les flux MJPEG).*

Si, dans le cas d'une ressource sans information temporelle, l'agent utilisateur sera néanmoins en mesure de rechercher un point antérieur à la première trame initialement fournie par le serveur, alors le temps zéro doit correspondre au premier temps de recherche du média [. ressource](#) ; sinon, il doit correspondre à la première trame reçue du serveur (le point de la [ressource média](#) auquel l'agent utilisateur a commencé à recevoir le flux).

*Au moment de la rédaction de cet article, aucun format connu ne manque de décalages de temps de trame explicites, mais prend toujours en charge la recherche d'une trame avant la première trame envoyée par le serveur.*

Considérez un flux d'un diffuseur de télévision, qui commence à diffuser un vendredi après-midi ensoleillé d'octobre, et envoie toujours aux agents utilisateurs connectés les données multimédias sur la même chronologie multimédia, avec son heure zéro définie sur le début de ce flux. Des mois plus tard, les agents utilisateurs se connectant à ce flux constateront que la première trame qu'ils reçoivent a une durée de plusieurs millions de secondes. La [getStartDate\(\)](#) méthode renvoie toujours la date à laquelle la diffusion a commencé ; cela permettrait aux contrôleurs d'afficher des temps réels dans leur épurateur (par exemple "14h30") plutôt qu'un temps relatif au début de la diffusion ("8 mois, 4 heures, 12 minutes et 23 secondes").

Considérez un flux qui transporte une vidéo avec plusieurs fragments concaténés, diffusé par un serveur qui ne permet pas aux agents utilisateurs de demander des heures spécifiques, mais diffuse simplement les données vidéo dans un ordre prédéterminé, la première image livrée étant toujours identifiée comme l'image avec temps zéro. Si un agent utilisateur se connecte à ce flux et reçoit des fragments définis comme couvrant les horodatages 2010-03-20 23:15:00 UTC à 2010-03-21 00:05:00 UTC et 2010-02-12 14:25:00 UTC au 12/02/2010 14:35:00 UTC, il exposerait cela avec une [chronologie des médias](#) commençant à 0s et s'étendant jusqu'à 3 600s (une heure). En supposant que le serveur de streaming se soit déconnecté à la fin du deuxième clip, l' [duration](#) attribut renverrait alors 3 600. La [getStartDate\(\)](#) méthode renverrait un [Date](#) objet avec une heure correspondant au 2010-03-20 23:15:00 UTC. Cependant, si un agent utilisateur différent se connectait cinq minutes plus tard, il recevrait (vraisemblablement) des fragments couvrant les horodatages 2010-03-20 23:20:00 UTC à 2010-03-21 00:05:00 UTC et 2010-02-12 14:25:00 UTC au 2010-02-12 14:35:00 UTC, et l'exposerait avec une [chronologie des médias](#) commençant à 0 s et s'étendant jusqu'à 3 300 s (cinquante-cinq minutes). Dans ce cas, la [getStartDate\(\)](#) méthode renverrait un [Date](#) objet avec une heure correspondant au 2010-03-20 23:20:00 UTC.

Dans ces deux exemples, l' [seekable](#) attribut donnerait les plages que le contrôleur voudrait réellement afficher dans son interface utilisateur ; typiquement, si les serveurs ne prennent pas en charge la recherche à des moments arbitraires, ce serait la plage de temps depuis le moment où l'agent utilisateur s'est connecté au flux jusqu'à la dernière trame que l'agent utilisateur a obtenue ; cependant, si l'agent utilisateur commence à ignorer les informations antérieures, la plage réelle peut être plus courte.

Dans tous les cas, l'agent utilisateur doit s'assurer que la [première position possible](#) (telle que définie ci-dessous) en utilisant la [chronologie des médias](#) établie , est supérieure ou égale à zéro.

La [chronologie des médias](#) a également une horloge associée. L'horloge utilisée est définie par l'agent utilisateur et peut dépendre [des ressources multimédias](#) , mais elle doit se rapprocher de l'horloge murale de l'utilisateur.

[Les éléments multimédias](#) ont une **position de lecture actuelle** , qui doit initialement (c'est-à-dire en l'absence de [données multimédias](#) ) être de zéro seconde. La [position de lecture actuelle](#) est une heure sur la [chronologie du média](#) .



[Les éléments multimédias](#) ont également une **position de lecture officielle** , qui doit initialement être définie sur zéro seconde. La [position de lecture officielle](#) est une approximation de la [position de lecture actuelle](#) qui est maintenue stable pendant l'exécution des scripts.

[Les éléments multimédias](#) ont également une **position de début de lecture par défaut** , qui doit initialement être définie sur zéro seconde. Ce temps permet de rechercher l'élément avant même le chargement du média.

Chaque [élément média](#) a un **drapeau d'affiche de spectacle** . Lorsqu'un [élément média](#) est créé, cet indicateur doit être défini sur vrai. Ce drapeau est utilisé pour contrôler quand l'agent utilisateur doit afficher une affiche pour un [video](#)élément au lieu d'afficher le contenu vidéo.

L' `currentTime`attribut doit, lors de l'obtention, renvoyer la [position de début de lecture par défaut](#) de l'[élément multimédia](#) , à moins qu'elle ne soit nulle, auquel cas il doit renvoyer la [position de lecture officielle](#) de l'élément . La valeur renvoyée doit être exprimée en secondes. Lors du réglage, si l' [élément multimédia](#) est , il doit définir la [position de début de lecture par défaut](#) de l' [élément multimédia](#) sur la nouvelle valeur ; sinon, il doit définir la [position de lecture officielle](#) sur la nouvelle valeur, puis [rechercher](#) la nouvelle valeur. La nouvelle valeur doit être interprétée comme étant en secondes.`readyStateHAVE NOTHING`

Si la [ressource multimédia](#) est une ressource de diffusion en continu, l'agent utilisateur peut être incapable d'obtenir certaines parties de la ressource après qu'elle ait expiré de sa mémoire tampon. De même, certaines [ressources multimédias](#) peuvent avoir une [chronologie multimédia](#) qui ne commence pas à zéro. La **première position possible** est la première position dans le flux ou la ressource que l'agent utilisateur peut jamais obtenir à nouveau. C'est aussi un moment sur la [chronologie des médias](#) .

*La [première position possible](#) n'est pas explicitement exposée dans l'API ; il correspond à l'heure de début de la première plage dans l' objet `seekable` de l'attribut `TimeRanges`, le cas échéant, ou à la [position de lecture actuelle](#) dans le cas contraire.*

Lorsque la [première position possible](#) change, alors : si la [position de lecture actuelle](#) est antérieure à la [première position possible](#) , l'agent utilisateur doit [rechercher](#) la [première position possible](#) ; sinon, si l'agent utilisateur n'a pas déclenché d' `timeupdate`événement sur l'élément au cours des 15 à 250 dernières ms et n'exécute pas encore de gestionnaires d'événements pour un tel événement, alors l'agent utilisateur doit mettre en file d'attente [une tâche d'élément multimédia](#) étant donné que l' [élément multimédia](#) déclenche un [événement](#) nommé `timeupdate`à l'élément.

*En raison de l'exigence ci-dessus et de l'exigence de l' [algorithme d'extraction de ressources](#) qui s'active [lorsque les métadonnées du clip sont connues](#) , la [position de lecture actuelle](#) ne peut jamais être inférieure à la [première position possible](#) .*

Si, à tout moment, l'agent utilisateur apprend qu'une piste audio ou vidéo est terminée et que toutes [les données multimédias](#) relatives à cette piste correspondent à des parties de la [chronologie multimédia](#) qui se trouvent *avant* la [première position possible](#), l'agent utilisateur peut [mettre en file d'attente une tâche d'élément multimédia](#) compte tenu de la [élément multimédia](#) pour exécuter ces étapes :

1. Supprimez la piste de l'objet [audioTracks](#) de l'attribut [AudioTrackList](#) ou de l'objet [videoTracks](#) de l'attribut [VideoTrackList](#), selon le cas.
2. [Lancez un événement](#) nommé [removetrack](#) sur l'objet ou mentionné ci-dessus de l'[élément multimédia](#), en utilisant, avec l'attribut initialisé sur l'objet ou représentant la piste. [AudioTrackListVideoTrackListTrackEventtrackAudioTrackVideoTrack](#)

L'[duration](#) attribut doit renvoyer l'heure de fin de la [ressource multimédia](#), en secondes, sur la [chronologie des médias](#). Si aucune [donnée multimédia](#) n'est disponible, les attributs doivent renvoyer la valeur Not-a-Number (NaN). Si la [ressource multimédia](#) n'est pas connue pour être limitée (par exemple, une radio en streaming ou un événement en direct sans heure de fin annoncée), l'attribut doit renvoyer la valeur Infinity positive.

L'agent utilisateur doit déterminer la durée de la [ressource multimédia](#) avant de lire une partie des [données multimédias](#) et avant de définir [readyState](#) une valeur égale ou supérieure à [HAVE\\_METADATA](#), même si cela nécessite de récupérer plusieurs parties de la ressource.

Lorsque la longueur de la [ressource média](#) passe à une valeur connue (par exemple, d'inconnue à connue, ou d'une longueur précédemment établie à une nouvelle longueur), l'agent utilisateur doit mettre en file d'attente [une tâche d'élément média](#) compte tenu de l'[élément média](#) pour [déclencher un événement](#) nommé [durationchange](#) à l'[élément média](#). (L'événement n'est pas déclenché lorsque la durée est réinitialisée dans le cadre du chargement d'une nouvelle ressource multimédia.) Si la durée est modifiée de sorte que la [position de lecture actuelle](#) finit par être supérieure à l'heure de la fin de la [ressource multimédia](#), alors l'utilisateur l'agent doit également [rechercher](#) l'heure de fin de la [ressource média](#).

Si un flux "infini" se termine pour une raison quelconque, la durée passera de l'infini positif à l'heure de la dernière image ou de l'échantillon du flux, et l'événement [durationchange](#) sera déclenché. De même, si l'agent utilisateur a initialement estimé la durée de [la ressource multimédia](#) au lieu de la déterminer avec précision, et révisé ensuite l'estimation en fonction de nouvelles informations, alors la durée changerait et l'[durationchange](#) événement serait déclenché.

Certains fichiers vidéo ont également une date et une heure explicites correspondant à l'heure zéro dans la [chronologie des médias](#), connue sous le nom **de décalage de la chronologie**. Initialement, le [décalage de la chronologie](#) doit être défini sur Not-a-Number (NaN).



La `getStartDate()` méthode doit renvoyer [un nouvel `Date` objet](#) représentant le [décalage de la chronologie](#) actuelle .

---

L' `loop` attribut est un [attribut booléen](#) qui, s'il est spécifié, indique que l' [élément média](#) doit revenir au début de la [ressource média](#) lorsqu'il atteint la fin.



L' `loop` attribut IDL doit [refléter](#) l'attribut de contenu du même nom.

#### 4.8.11.7 États prêts

`media.readyState`



Renvoie une valeur qui exprime l'état actuel de l'élément par rapport au rendu de la [position de lecture actuelle](#) , à partir des codes de la liste ci-dessous.

[Les éléments multimédias](#) ont un *état prêt* , qui décrit dans quelle mesure ils sont prêts à être rendus à la [position de lecture actuelle](#) . Les valeurs possibles sont les suivantes ; l'état prêt d'un élément multimédia à un moment donné est la plus grande valeur décrivant l'état de l'élément :

##### **HAVE\_NOTHING(valeur numérique 0)**

Aucune information concernant la [ressource média](#) n'est disponible. Aucune donnée pour la [position de lecture actuelle](#) n'est disponible. [Les éléments multimédias](#) dont `networkState` l'attribut est défini sur `NETWORK_EMPTY` sont toujours dans l' `HAVE_NOTHING` état .

##### **HAVE\_METADATA(valeur numérique 1)**

Une quantité suffisante de la ressource a été obtenue pour que la durée de la ressource soit disponible. Dans le cas d'un `video` élément, les dimensions de la vidéo sont également disponibles. Aucune [donnée multimédia](#) n'est disponible pour la [position de lecture actuelle](#) immédiate .

##### **HAVE\_CURRENT\_DATA(valeur numérique 2)**

Les données pour la [position de lecture actuelle](#) immédiate sont disponibles, mais soit il n'y a pas assez de données disponibles pour que l'agent utilisateur puisse avancer avec succès la [position de lecture actuelle](#) dans le [sens de la lecture](#) sans revenir immédiatement à l' `HAVE_METADATA` état, soit il n'y a plus de données à obtenir dans le [sens de la lecture](#) . Par exemple, en vidéo, cela

correspond à l'agent utilisateur ayant des données de l'image actuelle, mais pas de l'image suivante, lorsque la [position de lecture actuelle](#) est à la fin de l'image actuelle ; et jusqu'à la [fin de la lecture](#) .

#### **HAVE\_FUTURE\_DATA(valeur numérique 3)**

Les données pour la [position de lecture actuelle](#) immédiate sont disponibles, ainsi que suffisamment de données pour que l'agent utilisateur avance la [position de lecture actuelle](#) dans la [direction de lecture](#) au moins un peu sans revenir immédiatement à l' [HAVE\\_METADATA](#) état, et [les pistes de texte sont prêtes](#) . Par exemple, en vidéo, cela correspond à l'agent utilisateur ayant des données pour au moins l'image actuelle et l'image suivante lorsque la [position de lecture actuelle](#) est à l'instant entre les deux images, ou à l'agent utilisateur ayant les données vidéo pour la image actuelle et données audio pour continuer à jouer au moins un peu lorsque la [position de lecture actuelle](#) est au milieu d'un cadre. L'agent utilisateur ne peut pas être dans cet état si [la lecture est terminée](#) , car la [position de lecture actuelle](#) ne peut jamais avancer dans ce cas.

#### **HAVE\_ENOUGH\_DATA(valeur numérique 4)**

Toutes les conditions décrites pour l' [HAVE\\_FUTURE\\_DATA](#) état sont remplies et, en outre, l'une des conditions suivantes est également vraie :

- L'agent utilisateur estime que les données sont récupérées à un rythme où la [position de lecture actuelle](#) , si elle devait avancer au niveau de l'élément [playbackRate](#), ne dépasserait pas les données disponibles avant que la lecture n'atteigne la fin de la [ressource multimédia](#) .
- L'agent utilisateur est entré dans un état où attendre plus longtemps n'entraînera pas l'obtention de données supplémentaires, et donc rien ne serait gagné en retardant davantage la lecture. (Par exemple, la mémoire tampon peut être pleine.)

*En pratique, la différence entre [HAVE\\_METADATA](#) et [HAVE\\_CURRENT\\_DATA](#) est négligeable. Vraiment, la seule fois où la différence est pertinente, c'est lors de la peinture d'un [video](#) élément sur un [canvas](#), où elle distingue le cas où quelque chose sera dessiné ( [HAVE\\_CURRENT\\_DATA](#) ou plus) du cas où rien n'est dessiné ( [HAVE\\_METADATA](#) ou moins). De même, la différence entre [HAVE\\_CURRENT\\_DATA](#) (uniquement la trame courante) et [HAVE\\_FUTURE\\_DATA](#) (au moins cette trame et la suivante) peut être négligeable (à l'extrême, une seule trame). La seule fois où cette distinction compte vraiment, c'est lorsqu'une page fournit une interface pour la navigation "image par image".*

Lorsque l'état prêt d'un [élément multimédia](#) qui [networkState](#) n'est pas [NETWORK\\_EMPTY](#) change, l'agent utilisateur doit suivre les étapes ci-dessous :

1. Appliquez le premier ensemble de sous-étapes applicables de la liste suivante :

Si l'état prêt précédent était HAVE NOTHING, et que le nouvel état prêt est HAVE METADATA

Mettre en file d'attente une tâche d'élément multimédia en fonction de l' élément multimédia pour déclencher un événement nommé loadedmetadata au niveau de l'élément.

*Avant l'exécution de cette tâche, dans le cadre du mécanisme de boucle d'événements, le rendu aura été mis à jour pour redimensionner l'videoélément si nécessaire.*

Si l'état prêt précédent était HAVE METADATA et que le nouvel état prêt est HAVE CURRENT DATA ou supérieur

Si c'est la première fois que cela se produit pour cet élément média depuis load() la dernière invocation de l'algorithme, l'agent utilisateur doit mettre en file d'attente une tâche d'élément média donné l' élément média pour déclencher un événement nommé loadeddata à l'élément.

Si le nouvel état prêt est HAVE FUTURE DATA ou HAVE ENOUGH DATA, les étapes correspondantes ci-dessous doivent également être exécutées.

Si l'état prêt précédent était HAVE FUTURE DATA ou plus, et le nouvel état prêt est HAVE CURRENT DATA ou moins

Si l' élément multimédia était potentiellement en cours de lecture avant que son readyState attribut ne passe à une valeur inférieure à HAVE FUTURE DATA, et que l'élément n'a pas terminé la lecture, et que la lecture ne s'est pas arrêtée en raison d'erreurs, interrompue pour l'interaction de l'utilisateur ou interrompue pour le contenu intrabande, l'agent utilisateur doit mettre en file d'attente une tâche d'élément multimédia étant donné l' élément multimédia pour déclencher un événement nommé timeupdate à l'élément, et mettre en file d'attente une tâche d'élément multimédia étant donné l' élément multimédia pour déclencher un événement nommé waiting à l'élément.

Si l'état prêt précédent était HAVE CURRENT DATA ou moins, et que le nouvel état prêt est HAVE FUTURE DATA

L'agent utilisateur doit mettre en file d'attente une tâche d'élément multimédia donnée par l' élément multimédia pour déclencher un événement nommé canplay à l'élément.

Si l' paused attribut de l'élément est faux, l'agent utilisateur doit notifier la lecture de l'élément.

Si le nouvel état prêt est HAVE ENOUGH DATA

Si l'état prêt précédent était HAVE CURRENT DATA ou moins, l'agent utilisateur doit mettre en file d'attente une tâche d'élément multimédia donnée à l' élément multimédia pour déclencher un événement nommé canplay à

l'élément, et, si l' pausedattribut de l'élément est faux, notifier de la lecture pour l'élément.

L'agent utilisateur doit mettre en file d'attente une tâche d'élément multimédia donnée par l' élément multimédia pour déclencher un événement nommé canplaythrough à l'élément.

Si l'élément n'est pas éligible pour la lecture automatique , l'agent utilisateur doit abandonner ces sous-étapes.

L'agent utilisateur peut exécuter les sous-étapes suivantes :

1. Définissez l' pausedattribut sur false.
2. Si l' indicateur d'affiche d'exposition de l'élément est vrai, définissez-le sur faux et exécutez les marches temporelles par étapes.
3. Mettez en file d'attente une tâche d'élément multimédia donnée à l'élément pour déclencher un événement nommé play sur l'élément.
4. Avertir de jouer pour l'élément.

Alternativement, si l'élément est un videoélément, l'agent utilisateur peut commencer à observer si l'élément croise la fenêtre . Lorsque l'élément commence à croiser la fenêtre , si l'élément est toujours éligible pour la lecture automatique , exécutez les sous-étapes ci-dessus. Facultativement, lorsque l'élément cesse de croiser la fenêtre , si l' indicateur de lecture automatique est toujours vrai et que l' autoplayattribut est toujours spécifié, exécutez les sous-étapes suivantes :

5. Exécutez les étapes de pause internes et définissez l' indicateur can autoplay sur true.
6. Mettez en file d'attente une tâche d'élément multimédia donnée à l'élément pour déclencher un événement nommé pause sur l'élément.

*Les sous-étapes de lecture et de pause peuvent s'exécuter plusieurs fois lorsque l'élément démarre ou s'arrête en croisant la fenêtre , tant que l' indicateur de lecture automatique est vrai.*

*Les agents utilisateurs n'ont pas besoin de prendre en charge la lecture automatique, et il est suggéré que les agents utilisateurs respectent les préférences des utilisateurs en la matière. Les auteurs sont priés d'utiliser l' autoplayattribut plutôt que d'utiliser un script pour forcer la lecture de la vidéo, afin de permettre à l'utilisateur de remplacer le comportement s'il le souhaite.*

*Il est possible que l'état prêt d'un élément multimédia saute entre ces états de manière discontinue. Par exemple, l'état d'un élément média peut passer directement de HAVE METADATA à HAVE ENOUGH DATA sans passer par les états HAVE CURRENT DATA et HAVE FUTURE DATA.*

L' **readyState** attribut IDL doit, lors de l'obtention, renvoyer la valeur décrite ci-dessus qui décrit l'état prêt actuel de l' [élément média](#) .

L' **autoplay** attribut est un [attribut booléen](#) . Lorsqu'il est présent, l'agent utilisateur (tel que décrit dans l'algorithme décrit ici) commencera automatiquement la lecture de la [ressource multimédia](#) dès qu'il pourra le faire sans s'arrêter.

*Les auteurs sont priés d'utiliser l' **autoplay** attribut plutôt que d'utiliser un script pour déclencher la lecture automatique, car cela permet à l'utilisateur de remplacer la lecture automatique lorsqu'il n'est pas souhaité, par exemple lors de l'utilisation d'un lecteur d'écran. Les auteurs sont également encouragés à envisager de ne pas utiliser du tout le comportement de lecture automatique, et à la place de laisser l'agent utilisateur attendre que l'utilisateur démarre explicitement la lecture.*

✓ MDN

L' **autoplay** attribut IDL doit [refléter](#) l'attribut de contenu du même nom.

#### 4.8.11.8 Lecture de la ressource multimédia

**media.paused**

✓

Renvoie vrai si la lecture est en pause ; faux sinon.

**media.ended**

✓

Renvoie vrai si la lecture a atteint la fin de la [ressource multimédia](#) .

**media.defaultPlaybackRate** [ = value ]

✓

Renvoie le taux de lecture par défaut, lorsque l'utilisateur n'effectue pas d'avance ou de retour rapide dans la [ressource multimédia](#) .

Peut être défini pour modifier le taux de lecture par défaut.

Le taux par défaut n'a aucun effet direct sur la lecture, mais si l'utilisateur passe en mode d'avance rapide, lorsqu'il revient au mode de lecture normal, on s'attend à ce que le taux de lecture revienne au taux de lecture par défaut.

**media.playbackRate** [ = value ]

✓

Renvoie la vitesse de lecture actuelle, où 1,0 est la vitesse normale.

Peut être réglé, pour changer le taux de lecture.

**media.preservesPitch**

Renvoie vrai si des algorithmes préservant la hauteur tonale sont utilisés lorsque la valeur `playbackRate` n'est pas 1.0. La valeur par défaut est true.

Peut être défini sur false pour que la hauteur audio de la [ressource multimédia](#) change vers le haut ou vers le bas en fonction du fichier `playbackRate`. Ceci est utile pour des raisons esthétiques et de performance.

`media.played`

Renvoie un `TimeRanges` objet qui représente les plages de la [ressource multimédia](#) que l'agent utilisateur a lues.

`media.play()`

✓

Définit l' `paused` attribut sur false, en chargeant la [ressource multimédia](#) et en lançant la lecture si nécessaire. Si la lecture était terminée, la redémarrera depuis le début.

`media.pause()`

✓

Définit l' `paused` attribut sur true, en chargeant la [ressource multimédia](#) si nécessaire.

L' `paused` attribut indique si l' [élément multimédia](#) est en pause ou non. L'attribut doit initialement être vrai.

Un [élément multimédia](#) est un **élément multimédia bloqué** si son `readyState` attribut est dans l' `HAVE NOTHING` état, l' `HAVE METADATA` état ou l' `HAVE CURRENT DATA` état, ou si l'élément a été [mis en pause pour l'interaction de l'utilisateur](#) ou [mis en pause pour le contenu intrabande](#) .

On dit qu'un [élément multimédia](#) est **potentiellement en cours de lecture** lorsque son `paused` attribut est false, que l'élément n'a pas [terminé la lecture](#) , que la lecture ne s'est pas [arrêtée en raison d'erreurs](#) et que l'élément n'est pas un [élément multimédia bloqué](#) .

*Un `waiting` événement DOM [peut être déclenché](#) à la suite d'un [élément potentiellement en cours de lecture](#) qui arrête la lecture en raison `readyState` de la modification de son attribut par une valeur inférieure à `HAVE FUTURE DATA`.*

Un [élément multimédia](#) est considéré comme **éligible pour la lecture automatique** lorsque toutes les conditions suivantes sont remplies :

- Son [indicateur de lecture automatique](#) est vrai.
- Son `paused` attribut est vrai.
- Il a un `autoplay` attribut spécifié.

- [Le jeu d'indicateurs de sandboxing actif](#) de son [document de nœud](#) n'a pas le jeu [d'indicateurs de contexte de navigation des fonctionnalités automatiques en bac à sable](#) .
- Son [document de nœud](#) est [autorisé à utiliser](#) la [autoplay](#) fonctionnalité " " .

Un [élément multimédia](#) est dit être **autorisé à jouer** si l'agent utilisateur et le système autorisent la lecture multimédia dans le contexte actuel.

Par exemple, un agent utilisateur pourrait autoriser la lecture uniquement lorsque l'objet de [l'élément multimédia a une activation transitoire](#) , mais une exception pourrait être faite pour autoriser la lecture en mode [silencieux](#) [.Window](#)

On dit qu'un [élément multimédia a terminé sa lecture](#) lorsque :

- L' [readyState](#) attribut de l'élément est [HAVE METADATA](#) ou supérieur, et
- Soit:
  - La [position de lecture actuelle](#) est la fin de la [ressource multimédia](#) et
  - Le [sens de lecture](#) est vers l'avant, et
  - L' [élément média](#) n'a pas d' [loop](#) attribut spécifié.

Ou:

- La [position de lecture actuelle](#) est la [première position possible](#) , et
- Le [sens de lecture](#) est vers l'arrière.

L' [ended](#) attribut doit retourner vrai si, la dernière fois que la [boucle d'événement](#) a atteint [l'étape 1](#) , l' [élément média](#) avait [terminé la lecture](#) et que le [sens de lecture](#) était vers l'avant, et faux sinon.

On dit qu'un [élément multimédia](#) s'est **arrêté en raison d'erreurs** lorsque l' [readyState](#) attribut de l'élément est [HAVE METADATA](#) ou supérieur, et que l'agent utilisateur [rencontre une erreur non fatale](#) lors du traitement des [données multimédias](#) et, en raison de cette erreur, n'est pas en mesure de lire le contenu à la [position de lecture actuelle](#) .

On dit qu'un [élément multimédia](#) s'est mis en **pause pour l'interaction de l'utilisateur** lorsque son [paused](#) attribut est faux, l' [readyState](#) attribut est soit [HAVE FUTURE DATA](#) ou [HAVE ENOUGH DATA](#) et l'agent utilisateur a atteint un point dans la [ressource multimédia](#) où l'utilisateur doit faire une sélection pour que la ressource continue.

Il est possible qu'un [élément multimédia](#) ait à la fois [terminé la lecture](#) et [mis en pause pour l'interaction de l'utilisateur](#) .



Lorsqu'un [élément multimédia](#) qui est [potentiellement en cours de](#) lecture s'arrête parce qu'il a [fait une pause pour l'interaction de l'utilisateur](#), l'agent utilisateur doit [mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [timeupdate](#) sur l'élément.

On dit qu'un [élément multimédia](#) s'est **arrêté pour le contenu dans la bande** lorsque son [paused](#)attribut est faux, l' [readyState](#)attribut est soit [HAVE\\_FUTURE\\_DATA](#) ou [HAVE\\_ENOUGH\\_DATA](#) et l'agent utilisateur a suspendu la lecture de la [ressource multimédia](#) afin de lire le contenu qui est temporairement ancré à la [ressource multimédia](#) et a une longueur non nulle, ou pour lire un contenu qui est temporairement ancré à un segment de la [ressource multimédia](#) mais qui a une longueur supérieure à ce segment.

Un exemple de cas où un [élément multimédia](#) serait [mis en pause pour le contenu intrabande](#) est lorsque l'agent utilisateur lit [des descriptions audio](#) à partir d'un fichier WebVTT externe, et que la parole synthétisée générée pour un signal est plus longue que le temps entre l' [heure de début du signal de la piste de texte](#) et l' [heure de fin du repère de la piste de texte](#).

---

Lorsque la [position de lecture actuelle](#) atteint la fin de la [ressource multimédia](#) lorsque le [sens de lecture](#) est vers l'avant, l'agent utilisateur doit suivre ces étapes :

1. Si l' [élément média](#) a un [loop](#) attribut spécifié, alors [recherchez](#) la [première position possible](#) de la [ressource média](#) et revenez.
2. Comme défini ci-dessus, l' [ended](#)attribut IDL commence à renvoyer true une fois que la [boucle d'événements](#) revient à l'[étape 1](#).
3. [Mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) et des étapes suivantes :
  1. [Déclenchez un événement](#) nommé [timeupdate](#) au niveau de l' [élément média](#).
  2. Si l' [élément multimédia](#) a [terminé la lecture](#), le [sens de lecture](#) est vers l'avant et [pause](#) est faux, alors :
    1. Définissez l' [paused](#)attribut sur true.
    2. [Déclenchez un événement](#) nommé [pause](#) au niveau de l' [élément média](#).



3. [Prenez les promesses de jeu en attente](#) et [rejetez les promesses de jeu en attente](#) avec le résultat et un ["AbortError" DOMException](#).
3. [Déclenchez un événement](#) nommé [ended](#) au niveau de l' [élément média](#).

Lorsque la [position de lecture actuelle](#) atteint la [première position possible](#) de la [ressource multimédia](#) lorsque le [sens de lecture](#) est vers l'arrière, l'agent utilisateur doit uniquement [mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [timeupdate](#) à l'élément.

*Le mot "atteint" ici n'implique pas que la [position de lecture actuelle](#) doit avoir changé pendant la lecture normale ; cela pourrait être via [la recherche](#) , par exemple.*

---

L' [defaultPlaybackRate](#) attribut donne la vitesse souhaitée à laquelle la [ressource multimédia](#) doit jouer, en tant que multiple de sa vitesse intrinsèque. L'attribut est modifiable : lors de son obtention, il doit renvoyer la dernière valeur à laquelle il a été défini, ou 1,0 s'il n'a pas encore été défini ; lors de la définition, l'attribut doit être défini sur la nouvelle valeur.

*Le [defaultPlaybackRate](#) est utilisé par l'agent utilisateur lorsqu'il [expose une interface utilisateur à l'utilisateur](#).*

L' [playbackRate](#) attribut donne le taux de lecture effectif, qui est la vitesse à laquelle la [ressource multimédia](#) est lue, en tant que multiple de sa vitesse intrinsèque. S'il n'est pas égal à [defaultPlaybackRate](#), cela implique que l'utilisateur utilise une fonctionnalité telle que l'avance rapide ou la lecture au ralenti. L'attribut est modifiable : lors de son obtention, il doit renvoyer la dernière valeur à laquelle il a été défini, ou 1,0 s'il n'a pas encore été défini ; lors de la configuration, l'agent utilisateur doit suivre ces étapes :

1. Si la valeur donnée n'est pas prise en charge par l'agent utilisateur, lancez un ["NotSupportedError" DOMException](#).
2. Définissez [playbackRate](#) la nouvelle valeur et, si l'élément est [potentiellement en cours de lecture](#) , modifiez la vitesse de lecture.

Lorsque les attributs [defaultPlaybackRate](#) ou [playbackRate](#) changent de valeur (soit en étant définis par un script, soit en étant modifiés directement par l'agent utilisateur, par exemple en réponse au contrôle de l'utilisateur), l'agent utilisateur doit mettre en file d'attente [une tâche d'élément multimédia](#) étant donné l' [élément multimédia](#) pour [déclencher un événement](#) nommé [ratechange](#) au [élément](#)

[média](#) . L'agent utilisateur doit traiter les changements d'attributs sans à-coups et ne doit pas introduire d'intervalles perceptibles ni de coupure de lecture en réponse.

Les **preservesPitch** étapes du getter consistent à renvoyer true si un algorithme préservant la hauteur est en vigueur pendant la lecture. Les étapes de réglage consistent à activer ou désactiver en conséquence l'algorithme de préservation de la hauteur, sans aucun écart perceptible ni mise en sourdine de la lecture. Par défaut, un tel algorithme préservant la hauteur tonale doit être en vigueur (c'est-à-dire que le getter renverra initialement true).

---

L' **played** attribut doit renvoyer un nouvel [objet normalisé TimeRanges](#) statique qui représente les plages de points sur la [chronologie multimédia](#) de la [ressource multimédia](#) atteinte par l'augmentation monotone habituelle de la [position de lecture actuelle](#) pendant la lecture normale, le cas échéant, au moment où l'attribut est évalué.

***Retourner un nouvel objet à chaque fois est un mauvais modèle pour les getters d'attributs et n'est inscrit ici que car il serait coûteux de le changer. Il ne doit pas être copié dans de nouvelles API.***

---

Chaque [élément multimédia](#) a une **liste de promesses de lecture en attente** , qui doit initialement être vide.

Pour **accepter les promesses de lecture en attente** pour un [élément multimédia](#) , l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit *les promesses* une liste vide de promesses.
2. Copiez la [liste des promesses de lecture en attente](#) de [l'élément multimédia](#) dans *les promesses* .
3. Efface la [liste des promesses de lecture en attente](#) de [l'élément multimédia](#) .
4. *Promesses de retour* .

Pour **résoudre les promesses de lecture en attente** pour un [élément multimédia](#) avec une liste de promesses promises , l'agent utilisateur doit résoudre chaque promesse en *promises* avec undefined.

Pour **rejeter les promesses de lecture en attente** pour un [élément multimédia](#) avec une liste de promesses et une *erreur* de nom d'exception, l'agent utilisateur doit rejeter chaque promesse dans *les promesses avec erreur*.

Pour **notifier la lecture** d'un [élément multimédia](#), l'agent utilisateur doit exécuter les étapes suivantes :

1. [Prenez les promesses de jeu en attente](#) et laissez *les promesses* être le résultat.
2. [Mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l'élément et des étapes suivantes :
  1. [Lancez un événement](#) nommé `playing` à l'élément.
  2. [Résolvez les promesses de jeu en attente](#) avec *des promesses*.

Lorsque la `play()` méthode sur un [élément multimédia](#) est invoquée, l'agent utilisateur doit exécuter les étapes suivantes.

1. Si l' [élément média](#) n'est pas [autorisé à jouer](#), alors renvoyez [une promesse rejetée avec](#) un `"NotAllowedError" DOMException`.
2. Si l' attribut de l' [élément média](#)`error` n'est pas nul et que son [code](#) l' est `MEDIA_ERR_SRC_NOT_SUPPORTED`, alors renvoyez [une promesse rejetée avec](#) un `"NotSupportedError" DOMException`.

*Cela signifie que les [étapes d'échec de la source multimédia dédiée](#) ont été exécutées. La lecture n'est pas possible tant que l' [algorithme de chargement de l'élément multimédia](#) n'efface pas l' `error` attribut.*

3. Soit *promesse* une nouvelle promesse et ajoute *promesse* à la [liste des promesses de lecture en attente](#).
4. Exécutez les [étapes de lecture internes](#) pour l' [élément multimédia](#).
5. *Promesse de retour*.

Les **étapes de lecture internes** pour un [élément multimédia](#) sont les suivantes :

1. Si l' attribut de l' [élément média](#)`networkState` a la valeur `NETWORK_EMPTY`, appelez l' [algorithme de sélection](#) des ressources de l' [élément média](#).
2. Si la [lecture est terminée](#) et que le [sens de lecture](#) est vers l'avant, [recherchez](#) la [position la plus proche possible](#) de la [ressource multimédia](#).

*Cela [amènera](#) l'agent utilisateur à [mettre en file d'attente une tâche d'élément multimédia](#) étant donné que l' [élément multimédia](#) déclenche un [événement](#) nommé `timeupdate` à l' [élément multimédia](#).*

3. Si l'attribut de l' [élément média](#)`paused` est vrai, alors :
    1. Remplacez la valeur de `paused` par `false`.
    2. Si l' [indicateur d'affiche du spectacle](#) est vrai, définissez l' [indicateur d'affiche du spectacle](#) de l'élément sur faux et exécutez les [marches temporelles par](#) étapes.
    3. [Mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé `play` au niveau de l'élément.
    4. Si l'attribut de l' [élément média](#)`readyState` a la valeur `HAVE NOTHING`, `HAVE METADATA` ou `HAVE CURRENT DATA`, [mettez en file d'attente une tâche d'élément média](#) donnée à l' [élément média](#) pour [déclencher un événement](#) nommé `waiting` sur l'élément.  
  
Sinon, l'attribut de l' [élément média](#)`readyState` a la valeur `HAVE FUTURE DATA` ou `HAVE ENOUGH DATA`: [notifier de la lecture](#) pour l'élément.
  4. Sinon, si l'attribut de l' [élément média](#)`readyState` a la valeur `HAVE FUTURE DATA` ou `HAVE ENOUGH DATA`, [prenez les promesses de lecture en attente](#) et [mettez en file d'attente une tâche d'élément média](#) en fonction de l' [élément média](#) pour [résoudre les promesses de lecture en attente](#) avec le résultat.  
  

*L'élément multimédia est déjà en cours de lecture. Cependant, il est possible que la promesse soit [rejetée](#) avant l'exécution de la tâche en file d'attente.*
  5. Définissez l' [indicateur de lecture automatique](#) de l' [élément multimédia](#) sur `false`.
- 

Lorsque la `pause()` méthode est invoquée et lorsque l'agent utilisateur doit mettre en pause l' [élément média](#) , l'agent utilisateur doit exécuter les étapes suivantes :

1. Si l'attribut de l' [élément média](#)`networkState` a la valeur `NETWORK EMPTY`, appelez l' [algorithme de sélection](#) des ressources de l' [élément média](#) .
2. Exécutez les [étapes de pause interne](#) pour l' [élément multimédia](#) .

Les **étapes de pause interne** pour un [élément multimédia](#) sont les suivantes :

1. Définissez l' [indicateur de lecture automatique](#) de l' [élément multimédia](#) sur `false`.

2. Si l'attribut de l' [élément média](#) `paused` est faux, exécutez les étapes suivantes :
    1. Remplacez la valeur de `paused` par `true`.
    2. [Prenez les promesses de jeu en attente](#) et laissez *les promesses* être le résultat.
    3. [Mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) et des étapes suivantes :
      1. [Lancez un événement](#) nommé `timeupdate` à l'élément.
      2. [Lancez un événement](#) nommé `pause` à l'élément.
      3. [Rejeter les promesses de jeu en attente](#) avec *des promesses* et un `"AbortError" DOMException` .
    4. Réglez la [position de lecture officielle](#) sur la [position de lecture actuelle](#) .
- 

Si l'élément `playbackRate` est positif ou nul, alors la **direction de lecture** est vers l'avant. Sinon, c'est à l'envers.

Lorsqu'un [élément multimédia](#) est [potentiellement en cours de lecture](#) et qu'il `Document` est [entièrement actif](#) `Document` , sa [position de lecture actuelle](#) doit augmenter de manière monotone aux `playbackRate` unités de temps multimédia de l'élément par unité de temps de l' horloge de la [chronologie multimédia](#) . (Cette spécification se réfère toujours à cela comme une *augmentation* , mais cette augmentation pourrait en fait être une *diminution* si l'élément `playbackRate` est négatif.)

*L'élément `playbackRate` peut être 0.0, auquel cas la [position de lecture actuelle](#) ne bouge pas, même si la lecture n'est pas en pause ( `paused` ne devient pas vrai et l' `pause` événement ne se déclenche pas).*  
*Cette spécification ne définit pas comment l'agent utilisateur atteint le taux de lecture approprié - selon le protocole et les médias disponibles, il est plausible que l'agent utilisateur puisse négocier avec le serveur pour que le serveur fournisse les données multimédias au taux approprié, donc que (à l'exception de la période entre le moment où le débit est modifié et le moment où le serveur met à jour le débit de lecture du flux) le client n'a pas réellement besoin de supprimer ou d'interpoler des images.*

Chaque fois que l'agent utilisateur [fournit un état stable](#) , la [position de lecture officielle](#) doit être définie sur la [position de lecture actuelle](#) .

Lorsque le [sens de lecture](#) est inversé, tout son correspondant doit être [mis en sourdine](#) . Tant que l'élément [playbackRate](#) est si faible ou si élevé que l'agent utilisateur ne peut pas lire l'audio de manière utile, l'audio correspondant doit également être [mis en sourdine](#) . Si l'élément [playbackRate](#) n'est pas 1.0 et [preservesPitch](#) est vrai, l'agent utilisateur doit appliquer un ajustement de hauteur pour préserver la hauteur originale de l'audio. Sinon, l'agent utilisateur doit accélérer ou ralentir l'audio sans aucun ajustement de hauteur.

Lorsqu'un [élément multimédia](#) est [potentiellement en cours de lecture](#) , ses données audio lues doivent être synchronisées avec la [position de lecture actuelle](#) , au [volume multimédia effectif](#) de l'élément . L' agent utilisateur doit lire l' audio des pistes audio qui ont été activées lorsque la [boucle d' événements](#) a atteint l' [étape 1](#) pour la dernière fois .

Lorsqu'un [élément multimédia](#) n'est pas [potentiellement en cours de lecture](#) , l'audio ne doit pas être lu pour l'élément.

[Les éléments multimédias](#) qui sont [potentiellement en cours de lecture](#) alors qu'ils ne se trouvent pas [dans un document](#) ne doivent pas lire de vidéo, mais doivent lire n'importe quel composant audio. Les éléments multimédias ne doivent pas s'arrêter de jouer simplement parce que toutes les références à ceux-ci ont été supprimées ; ce n'est qu'une fois qu'un élément multimédia est dans un état où plus aucun son ne peut plus être lu par cet élément que l'élément peut être ramassé.

*Il est possible qu'un élément auquel aucune référence explicite n'existe puisse lire de l'audio, même si un tel élément n'est pas encore en cours de lecture : par exemple, il peut être rétabli mais bloqué en attendant que le contenu soit mis en mémoire tampon, ou il peut être encore en mémoire tampon, mais avec un [suspend](#) écouteur d'événement qui commence la lecture. Même un élément multimédia dont [la ressource multimédia](#) n'a pas de pistes audio pourrait éventuellement rejouer l'audio s'il avait un écouteur d'événement qui modifie la [ressource multimédia](#) .*

---

Chaque [élément multimédia](#) a une **liste de nouvelles cues introduites** , qui doit être initialement vide. Chaque fois qu'une [cue de piste de texte](#) est ajoutée à la [liste des cues](#) d'une [piste de texte](#) qui se trouve dans la [liste des pistes de texte](#) d'un [élément média](#) , cette [cue](#) doit être ajoutée à la liste des [cues nouvellement introduites de l'élément média](#) . Chaque fois qu'une [piste de texte](#) est ajoutée à la [liste des pistes de texte](#) pour un [élément multimédia](#) , toutes les [répliques de la liste des répliques](#) de cette [piste de texte](#) doivent être ajoutés à la [liste des nouveaux indices introduits par](#) l' élément [média](#) . Lorsque la liste des [indices nouvellement introduits d' un élément multimédia](#) contient de nouveaux indices ajoutés alors que l' [indicateur d'affiche d'affichage](#) de l'[élément multimédia](#) n'est pas défini, l'agent utilisateur doit exécuter les [marches temporelles par](#) étapes.



Lorsqu'une [cue de piste de texte](#) est supprimée de la [liste des cues](#) d'une [piste de texte](#) qui se trouve dans la [liste des pistes de texte](#) d'un [élément multimédia](#) , et chaque fois qu'une [piste de texte](#) est supprimée de la [liste des pistes de texte](#) d'un [élément multimédia](#) , si le [média](#) l'[indicateur d'affichage](#) de l'[élément](#) n'est pas défini, l'agent utilisateur doit exécuter les [marches temporelles par](#) étapes.

Lorsque la [position de lecture actuelle](#) d'un [élément multimédia](#) change (par exemple en raison de la lecture ou de la recherche), l'agent utilisateur doit exécuter les [marches temporelles par](#) étapes. Pour prendre en charge les cas d'utilisation qui dépendent de la précision temporelle du déclenchement des événements de repère, tels que la synchronisation des sous-titres avec les changements de plan dans une vidéo, les agents utilisateurs doivent déclencher les événements de repère aussi près que possible de leur position sur la chronologie du média, et idéalement dans les 20 millisecondes. Si la [position de lecture actuelle](#) change pendant l'exécution des étapes, l'agent utilisateur doit attendre la fin des étapes, puis doit immédiatement les réexécuter. Ces étapes sont donc exécutées aussi souvent que possible ou nécessaire.

*Si une itération prend beaucoup de temps, cela peut entraîner le saut [de signaux](#) de courte durée lorsque l'agent utilisateur se précipite pour "rattraper son retard", de sorte que ces signaux n'apparaîtront pas dans la [activeCues](#) liste.*

Les ***marches du temps sur*** les marches sont les suivantes :

1. Soit *les cues actuels* une liste de [cues](#) , initialisée pour contenir toutes les [cues](#) de toutes les [pistes de texte masquées](#) ou [affichées](#) de l' [élément multimédia](#) (pas celles [désactivées](#) ) dont [les heures de début](#) sont inférieures ou égales à la [position de lecture actuelle](#) et dont [les heures de fin](#) sont supérieures à la [position de lecture actuelle](#) .
2. Soit *other cues* une liste de [cues](#) , initialisée pour contenir toutes les [cues des pistes](#) de texte [cachées](#) et [affichées](#) de l' [élément média](#) qui ne sont pas présentes dans *les cues actuelles* .
3. Soit *dernière fois* la [position de lecture actuelle](#) au moment où cet algorithme a été exécuté pour la dernière fois pour cet [élément multimédia](#) , si ce n'est pas la première fois qu'il s'exécute.
4. Si la [position de lecture actuelle](#) n'a, depuis la dernière fois que cet algorithme a été exécuté, changé que par son augmentation monotone habituelle pendant la lecture normale, alors laissez les *cues manquées* être la liste des [cues](#) dans *d'autres cues* dont [les heures de début](#) sont supérieures ou égales à *la dernière fois* et dont [les heures de fin](#) sont inférieures ou égales à la [position de lecture actuelle](#) . Sinon, laissez *les repères manqués* être une liste vide.
5. Supprimez toutes les [cues](#) dans *les cues manquées* qui se trouvent également dans la [liste des cues nouvellement introduites](#) de l'[élément multimédia](#) , puis videz la [liste des cues nouvellement introduites](#) de l'élément .

6. Si le temps a été atteint par l'augmentation monotone habituelle de la [position de lecture actuelle](#) pendant la lecture normale, et si l'agent utilisateur n'a pas déclenché d' [timeupdate](#) événement sur l'élément au cours des 15 à 250 dernières ms et n'exécute pas encore de gestionnaires d'événements pour un tel événement, alors l'agent utilisateur doit [mettre en file d'attente une tâche d'élément média](#) donné l' [élément média](#) pour [déclencher un événement](#) nommé [timeupdate](#) à l'élément. (Dans les autres cas, tels que les recherches explicites, les événements pertinents sont déclenchés dans le cadre du processus global de modification de la [position de lecture actuelle](#) .)

*L'événement ne doit donc pas être déclenché plus rapidement qu'environ 66 Hz ou plus lent que 4 Hz (en supposant que les gestionnaires d'événements ne prennent pas plus de 250 ms pour s'exécuter). Les agents utilisateurs sont encouragés à faire varier la fréquence de l'événement en fonction de la charge du système et du coût moyen de traitement de l'événement à chaque fois, afin que les mises à jour de l'interface utilisateur ne soient pas plus fréquentes que ce que l'agent utilisateur peut gérer confortablement lors du décodage de la vidéo.*

7. Si toutes les [cues](#) des *cues actuelles* ont leur [drapeau actif de cue de piste de texte](#) défini, aucune des [cues](#) des *autres cues* n'a son [drapeau actif de cue de piste de texte](#) défini, et les *cues manquées* sont vides, puis revenez.
8. Si le temps a été atteint par l'augmentation monotone habituelle de la [position de lecture actuelle](#) pendant la lecture normale, et qu'il y a [des cues](#) dans *d'autres cues* qui ont leur [indicateur de pause à la sortie de la cue de la piste de texte](#) et qui ont leur [indicateur actif de cue de la piste de texte](#) défini ou sont également dans *des repères manqués* , puis mettez [immédiatement en pause](#) l' [élément multimédia](#) .

*Dans les autres cas, tels que les recherches explicites, la lecture n'est pas interrompue en dépassant l'heure de fin d'une [mémoire](#) , même si cette [mémoire](#) a son [indicateur de pause à la sortie de la mémoire de piste de texte](#) .*

9. Soit *events* une liste de [tâches](#) , initialement vide. Chaque [tâche](#) de cette liste sera associée à une [piste de texte](#) , un [repère de piste de texte](#) et une heure, qui sont utilisés pour trier la liste avant que les [tâches](#) ne soient mises en file d'attente.

Soit *les pistes affectées* une liste de [pistes de texte](#) , initialement vide.

Lorsque les étapes ci-dessous indiquent de **préparer un événement** nommé *event* pour une *cible* [de repère de piste de texte](#) avec un temps *time* , l'agent utilisateur doit exécuter ces étapes :

1. Soit *track* la [piste de texte](#) à laquelle la *cible de repère* [de la piste de texte](#) est associée.
2. Créez une [tâche](#) pour [déclencher un événement](#) nommé *event* sur *target* .



3. [Ajoutez la tâche](#) nouvellement créée aux *événements* , associée à l'heure *heure* , la *piste* [de piste de texte](#) et la *cible* [de repère de la piste de texte](#) .
  4. Ajouter *une piste* aux *pistes affectées* .
  10. Pour chaque [repère de piste de texte](#) dans *les repères manqués* , [préparez un événement](#) nommé [enter](#) pour l' [TextTrackCue](#) objet avec l' [heure de début du repère de piste de texte](#) .
  11. Pour chaque [cue de piste de texte](#) dans *d'autres cues* dont l'[indicateur actif de cue de piste de texte](#) est défini ou qui est dans *des cues manquées* , [préparez un événement](#) nommé [exit](#) pour l' [TextTrackCue](#) objet avec la plus tardive de l' [heure de fin de la cue de la piste de texte](#) et de l' [heure de début de la cue de la piste de texte](#) .
  12. Pour chaque [repère de piste de texte](#) dans *les repères actuels* dont l'[indicateur actif de repère de piste de texte](#) n'est pas défini , [préparez un événement](#) nommé [enter](#) pour l' [TextTrackCue](#) objet avec l' [heure de début du repère de piste de texte](#) .
  13. Triez les [tâches](#) dans *les événements* par ordre chronologique croissant ( [les tâches](#) avec des heures antérieures en premier).
- Triez davantage [les tâches](#) dans *les événements* qui ont la même heure selon l' [ordre relatif des repères de piste de texte](#) des [repères de piste de texte](#) associés à ces [tâches](#) .
- Enfin, triez [les tâches](#) dans *les événements* qui ont la même heure et le même [ordre de repère de piste de texte](#) en plaçant [les tâches](#) qui déclenchent [enter](#) des événements avant celles qui déclenchent [exit](#) des événements.
14. [Mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour chaque [tâche](#) dans *events* , dans l'ordre de la liste.
  15. Triez *les pistes concernées* dans le même ordre que les [pistes de texte](#) apparaissant dans la [liste des pistes de texte](#) de l'[élément multimédia](#) et supprimez les doublons.
  16. Pour chaque [piste de texte](#) dans *les pistes affectées* , dans l'ordre de la liste, [mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [cuechange](#) sur l' [TextTrack](#) objet et, si la [piste de texte](#) [track](#) a un élément correspondant , pour [déclencher ensuite un événement](#) nommé [cuechange](#) sur l'objet [track](#) élément également.
  17. Définissez le [drapeau actif de cue de piste de texte](#) de toutes les [cues](#) dans les *cues actuelles* et désactivez le [drapeau actif de cue de piste de texte](#) de toutes les [cues](#) dans les *autres cues* .

18. Exécutez les [règles de mise à jour du rendu de la piste de texte](#) de chacune des [pistes de texte](#) dans *les pistes affectées* qui [affichent](#) , en fournissant la [langue de la piste de texte](#) de la piste de texte comme langue de secours si ce n'est pas la chaîne vide. Par exemple, pour [les pistes de texte](#) basées sur WebVTT, les [règles de mise à jour de l'affichage des pistes de texte](#) [WebVTT](#) . [\[WEBVTT\]](#)

Pour les besoins de l'algorithme ci-dessus, un [signal de piste de texte](#) est considéré comme faisant partie d'une [piste de texte](#) uniquement s'il est répertorié dans la [liste des signaux de piste de texte](#) , et pas simplement s'il est associé à la [piste de texte](#) .

*Si le [document de nœud](#) de [l'élément multimédia](#) cesse d'être un document [entièrement actif](#) , la lecture s'arrêtera [jusqu'à](#) ce que le document soit à nouveau actif.*

Lorsqu'un [élément multimédia](#) est [supprimé d'un Document](#) , l'agent utilisateur doit exécuter les étapes suivantes :

1. [Attendez un état stable](#) , permettant à la [tâche](#) qui a supprimé l' [élément multimédia](#) du de [Document](#) se poursuivre. La [section synchrone](#) comprend toutes les étapes restantes de cet algorithme. (Les étapes de la [section synchrone](#) sont marquées d'un ⌚.)
2. ⌚ Si l' [élément média](#) est [dans un document](#) , retour.
3. ⌚ Exécutez les [étapes de pause interne](#) pour l' [élément multimédia](#) .

#### 4.8.11.9 Recherche

**`media. seeking`**

Renvoie true si l'agent utilisateur recherche actuellement.

**`media. seekable`**

✓ 

Renvoie un [TimeRanges](#) objet qui représente les plages de la [ressource multimédia](#) dans laquelle il est possible pour l'agent utilisateur de rechercher.

**`media. fastSeek (time)`**



Cherche à se rapprocher le plus rapidement possible de *l'heure donnée, en échangeant la précision contre la vitesse*. (Pour chercher à une heure précise, utilisez l' [currentTime](#) attribut.)

Cela ne fait rien si la ressource média n'a pas été chargée.

L' **seeking** attribut doit initialement avoir la valeur false.

La méthode doit **rechercher** le temps donné par *time* , avec l' indicateur *approximatif pour la vitesse* défini. **fastSeek (time)**

Lorsque l'agent utilisateur doit **rechercher** une *nouvelle position de lecture* particulière dans la **ressource multimédia** , éventuellement avec l' indicateur *approximatif de vitesse* défini, cela signifie que l'agent utilisateur doit exécuter les étapes suivantes. Cet algorithme interagit étroitement avec le mécanisme **de boucle d'événements** ; en particulier, il a une **section synchrone** (qui est déclenchée dans le cadre de l' algorithme **de boucle d'événement** ). Les étapes de cette section sont marquées d'un ⌚.

1. Définissez l' **indicateur d'affiche** de **l'élément multimédia** sur false.
2. Si l' **élément média** est , retournez **readyState .HAVE NOTHING**
3. Si l' **seeking**attribut IDL de l'élément est vrai, alors une autre instance de cet algorithme est déjà en cours d'exécution. Abandonnez cette autre instance de l'algorithme sans attendre la fin de l'étape en cours d'exécution.
4. Définissez l' **seeking**attribut IDL sur true.
5. Si la recherche était en réponse à un appel de méthode DOM ou à la définition d'un attribut IDL, continuez le script. Le reste de ces étapes doit être exécuté **en parallèle** . À l'exception des étapes marquées d'un ⌚, elles peuvent être interrompues à tout moment par une autre instance de cet algorithme invoquée.
6. Si la *nouvelle position de lecture* est postérieure à la fin de la **ressource multimédia** , laissez-la plutôt être la fin de la **ressource multimédia** .
7. Si la *nouvelle position de lecture* est inférieure à la **première position possible** , laissez-la être cette position à la place.
8. Si la *nouvelle position de lecture* (éventuellement maintenant modifiée) n'est pas dans l'une des plages données dans l' **seekable**attribut, alors que ce soit la position dans l'une des plages données dans l' **seekable**attribut qui est la plus proche de la *nouvelle position de lecture* . Si deux positions satisfont toutes les deux à cette contrainte (c'est-à-dire que la *nouvelle position de lecture* est exactement au milieu entre deux plages dans l' **seekable**attribut), utilisez alors la position la plus proche de la **position de lecture actuelle** . S'il n'y a pas de plages données dans l' **seekable**attribut, définissez l' **seeking**attribut IDL sur faux et retournez.
9. Si l' indicateur *d'approximation de la vitesse* est défini, réglez la *nouvelle position de lecture* sur une valeur qui permettra à la lecture de reprendre rapidement. Si la *nouvelle position de lecture* avant cette étape est avant la **position de lecture actuelle** , alors la *nouvelle position de lecture* ajustée doit également être avant la **position de lecture actuelle** . De même, si la *nouvelle*

*position de lecture* avant cette étape est après [la position de lecture actuelle](#) , alors la *nouvelle position de lecture* ajustée doit également être après la [position de lecture actuelle](#) .

Par exemple, l'agent utilisateur pourrait s'accrocher à une image clé à proximité, de sorte qu'il n'ait pas à passer du temps à décoder puis à supprimer les images intermédiaires avant de reprendre la lecture.

10. [Mettre en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [seeking](#) au niveau de l'élément.

11. Réglez la [position de lecture actuelle](#) sur la *nouvelle position de lecture* .

*Si l' [élément multimédia](#) était [potentiellement en cours de lecture](#) juste avant de commencer la recherche, mais que la recherche a fait [readyState](#) passer son attribut à une valeur inférieure à [HAVE FUTURE DATA](#), alors un [waiting](#) événement sera déclenché sur l'élément.*

*Cette étape définit la [position de lecture actuelle](#) et peut donc déclencher immédiatement d'autres conditions, telles que les règles concernant le moment où la lecture " [atteint la fin de la ressource multimédia](#) " (partie de la logique qui gère la boucle), avant même que l'agent utilisateur ne soit réellement capable pour restituer les données multimédias pour cette position (comme déterminé à l'étape suivante).*

*L' [currentTime](#) attribut renvoie la [position de lecture officielle](#) , et non la [position de lecture actuelle](#) , et est donc mis à jour avant l'exécution du script, indépendamment de cet algorithme.*

12. Attendez que l'agent utilisateur ait établi si oui ou non les [données multimédias](#) pour la *nouvelle position de lecture* sont disponibles et, si c'est le cas, jusqu'à ce qu'il ait décodé suffisamment de données pour lire cette position.

13. [Attendez un état stable](#) . La [section synchrone](#) comprend toutes les étapes restantes de cet algorithme. (Les étapes de la [section synchrone](#) sont marquées d'un ⌚.)

14. ⌚ Définissez l' [seeking](#) attribut IDL sur faux.

15. ⌚ Courez le [temps marche sur](#) les marches.

16. ⌚ [Mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [timeupdate](#) sur l'élément.

17. ⌚ [Mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour [déclencher un événement](#) nommé [seeked](#) sur l'élément.

---

L' **seekable** attribut doit renvoyer un nouvel [objet normalisé TimeRanges](#) statique qui représente les plages de la [ressource média](#) , s'il y en a, que l'agent utilisateur est capable de rechercher, au moment où l'attribut est évalué.

*Si l'agent utilisateur peut rechercher n'importe où dans la [ressource multimédia](#) , par exemple parce qu'il s'agit d'un simple fichier vidéo et que l'agent utilisateur et le serveur prennent en charge les requêtes HTTP Range, l'attribut renverra un objet avec une plage, dont le début est l'heure de la première image (la [position la plus ancienne possible](#) , généralement zéro) et dont la fin est identique à l'heure de la première image plus la [duration](#) valeur de l'attribut (qui serait égale à l'heure de la dernière image et pourrait être positive Infinity).*

*La plage peut changer continuellement, par exemple si l'agent utilisateur met en mémoire tampon une fenêtre glissante sur un flux infini. C'est le comportement observé avec les DVR qui regardent la télévision en direct, par exemple.*

***Retourner un nouvel objet à chaque fois est un mauvais modèle pour les getters d'attributs et n'est inscrit ici que car il serait coûteux de le changer. Il ne doit pas être copié dans de nouvelles API.***

Les agents utilisateurs devraient adopter une vision très libérale et optimiste de ce qui est rechercherable. Les agents utilisateurs doivent également mettre en mémoire tampon le contenu récent lorsque cela est possible pour permettre une recherche rapide.

Par exemple, considérez un fichier vidéo volumineux servi sur un serveur HTTP sans prise en charge des requêtes HTTP Range. Un navigateur *pourrait* implémenter cela en ne mettant en mémoire tampon que l'image actuelle et les données obtenues pour les images suivantes, n'autorise jamais la recherche, sauf pour la recherche au tout début en redémarrant la lecture. Cependant, ce serait une mauvaise mise en œuvre. Une implémentation de haute qualité mettrait en mémoire tampon les dernières minutes de contenu (ou plus, si un espace de stockage suffisant est disponible), permettant à l'utilisateur de revenir en arrière et de revoir quelque chose de surprenant sans aucune latence, et permettrait en outre une recherche arbitraire en rechargeant le fichier à partir de le démarrage si nécessaire, ce qui serait plus lent mais toujours plus pratique que de devoir littéralement redémarrer la vidéo et de la regarder tout au long juste pour arriver à un endroit antérieur sans tampon.

[Les ressources multimédias](#) peuvent être scénarisées en interne ou interactives. Ainsi, un [élément multimédia](#) pourrait jouer de manière non linéaire. Si cela se produit, l'agent utilisateur doit agir comme si l'algorithme de [recherche](#) était utilisé chaque fois que la [position de lecture actuelle](#) change de manière discontinue (de sorte que les événements pertinents se déclenchent).

#### 4.8.11.10 Ressources multimédias avec plusieurs pistes multimédias

Une [ressource multimédia](#) peut avoir plusieurs pistes audio et vidéo intégrées. Par exemple, en plus des pistes vidéo et audio principales, une [ressource multimédia](#) peut avoir des dialogues doublés en langue étrangère, des commentaires du réalisateur, des descriptions audio, des angles alternatifs ou des superpositions en langue des signes.

#### **media.audioTracks**

✓

Renvoie un [AudioTrackList](#) objet représentant les pistes audio disponibles dans la [ressource multimédia](#) .

#### **media.videoTracks**

✓

Renvoie un [VideoTrackList](#) objet représentant les pistes vidéo disponibles dans la [ressource multimédia](#) .

L' **audioTracks** attribut d'un [élément média](#) doit renvoyer un objet [live AudioTrackList](#) représentant les pistes audio disponibles dans la [ressource média](#) de l' [élément média](#) .

L' **videoTracks** attribut d'un [élément média](#) doit renvoyer un objet [live VideoTrackList](#) représentant les pistes vidéo disponibles dans la [ressource média](#) de l' [élément média](#) .

*Il n'y a jamais qu'un [AudioTrackList](#) objet et un [VideoTrackList](#) objet par [élément média](#) , même si une autre [ressource média](#) est chargée dans l'élément : les objets sont réutilisés. ( Cependant, les objets [AudioTrack](#) et ne le sont pas.)[VideoTrack](#)*

#### **4.8.11.10.1 [AudioTrackList](#) et [VideoTrackList](#) objets**

✓

MDN

Les interfaces [AudioTrackList](#) et [VideoTrackList](#) sont utilisées par les attributs définis dans la section précédente.

✓

MDN

[Exposed=Window]

```
interface AudioTrackList : EventTarget {
```

```
    readonly attribute unsigned long length;
```

```
getter AudioTrack (unsigned long index);
```

```
AudioTrack? getTrackById(DOMString id);
```

```
attribute EventHandler onchange;
```

```
attribute EventHandler onaddtrack;
```

```
attribute EventHandler onremovetrack;
```

```
};
```

```
[Exposed=Window]
```

```
interface AudioTrack {
```

```
readonly attribute DOMString id;
```

```
readonly attribute DOMString kind;
```

```
readonly attribute DOMString label;
```

```
readonly attribute DOMString language;
```

```
attribute boolean enabled;
```

```
};
```

```
[Exposed=Window]
```

```
interface VideoTrackList : EventTarget {
```

```
readonly attribute unsigned long length;
```

```
getter VideoTrack (unsigned long index);
```

```
VideoTrack? getTrackById(DOMString id);
```

```
readonly attribute long selectedIndex;
```

```
attribute EventHandler onchange;
```

```
attribute EventHandler onaddtrack;
```

```
attribute EventHandler onremovetrack;
```

```
};
```

```
[Exposed=Window]
```

```
interface VideoTrack {
```

```
  readonly attribute DOMString id;
```

```
  readonly attribute DOMString kind;
```

```
  readonly attribute DOMString label;
```

```
  readonly attribute DOMString language;
```

```
  attribute boolean selected;
```

```
};
```

**media.audioTracks.length**

✓ 

**media.videoTracks.length**

✓ 

Renvoie le nombre de pistes dans la liste.

```
audioTrack = media.audioTracks[index]
```

```
videoTrack = media.videoTracks[index]
```

Renvoie l'objet [AudioTrack](#) ou spécifié [VideoTrack](#).

```
audioTrack = media.audioTracks.getTrackById(id)
```

✓ 

```
videoTrack = media.videoTracks.getTrackById(id)
```

✓ 

Renvoie l'objet [AudioTrack](#) ou [VideoTrack](#) avec l'identifiant donné, ou null si aucune piste n'a cet identifiant.

**audioTrack.id**

✓ 



`videoTrack.id`

✓

Renvoie l'ID de la piste donnée. Il s'agit de l'ID qui peut être utilisé avec un [fragment](#) si le format prend en charge [la syntaxe de fragment de média](#) et qui peut être utilisé avec la `getTrackById()` méthode.

`audioTrack.kind`

✓

`videoTrack.kind`

✓

Renvoie la catégorie à laquelle appartient la piste donnée. Les [catégories de pistes possibles](#) sont indiquées ci-dessous.

`audioTrack.label`

✓

`videoTrack.label`

✓

Renvoie le label de la piste donnée, s'il est connu, ou la chaîne vide sinon.

`audioTrack.language`

✓

`videoTrack.language`

✓

Renvoie la langue de la piste donnée, si elle est connue, ou la chaîne vide sinon.

`audioTrack.enabled [ = value ]`

✓

Renvoie true si la piste donnée est active, et false sinon.

Peut être réglé, pour changer si la piste est activée ou non. Si plusieurs pistes audio sont activées simultanément, elles sont mixées.

`media.videoTracks.selectedIndex`

✓

Renvoie l'index de la piste actuellement sélectionnée, le cas échéant, ou -1 sinon.

`videoTrack.selected [ = value ]`

✓

Renvoie true si la piste donnée est active, et false sinon.

Peut être réglé, pour changer si la piste est sélectionnée ou non. Soit zéro soit une piste vidéo est sélectionnée ; sélectionner une nouvelle piste alors qu'une précédente est sélectionnée désélectionnera la précédente.

Un [AudioTrackList](#) objet représente une liste dynamique de zéro ou plusieurs pistes audio, dont zéro ou plusieurs peuvent être activées à la fois. Chaque piste audio est représentée par un [AudioTrack](#) objet.

Un [VideoTrackList](#) objet représente une liste dynamique de zéro ou plusieurs pistes vidéo, dont zéro ou une peut être sélectionnée à la fois. Chaque piste vidéo est représentée par un [VideoTrack](#) objet.

Les pistes [AudioTrackList](#) et [VideoTrackList](#) les objets doivent être ordonnés de manière cohérente. Si la [ressource multimédia](#) est dans un format qui définit un ordre, alors cet ordre doit être utilisé ; sinon, l'ordre doit être l'ordre relatif dans lequel les pistes sont déclarées dans la [ressource média](#) . L'ordre utilisé est appelé l' *ordre naturel* de la liste.

*Chaque piste de l'un de ces objets possède donc un index ; la première a l'indice 0, et chaque piste suivante est numérotée une de plus que la précédente. Si une [ressource multimédia](#) ajoute ou supprime dynamiquement des pistes audio ou vidéo, les indices des pistes changeront de manière dynamique. Si la [ressource multimédia](#) change entièrement, toutes les pistes précédentes seront supprimées et remplacées par de nouvelles pistes.*

Les getters d'attribut [AudioTrackList length](#) et [VideoTrackList length](#) doivent renvoyer le nombre de pistes représentées par leurs objets au moment de l'obtention.

Les [indices de propriété pris en charge](#) des objets [AudioTrackList](#) et [VideoTrackList](#) à tout instant sont les nombres de zéro au nombre de pistes représentées par l'objet respectif moins un, si des pistes sont représentées. Si un objet [AudioTrackList](#) ou [VideoTrackList](#) ne représente aucune piste, il n'a pas [d'indices de propriété pris en charge](#) .

Pour [déterminer la valeur d'une propriété indexée](#) pour un *index* index donné dans une *liste* d'objets [AudioTrackList](#) ou [VideoTrackList](#), l'agent utilisateur doit renvoyer l' objet or qui représente la piste *d'index* ème dans la *liste* .  
[VideoTrackListAudioTrackVideoTrack](#)

Les méthodes [getTrackById\(id\)](#) doivent retourner le premier objet ou (respectivement) dans l' objet ou (respectivement) dont l'identifiant est égal à la valeur de l' argument *id* (dans l'ordre naturel de la liste, tel que défini ci-dessus). Lorsqu'aucune piste ne correspond à l'argument donné, les méthodes doivent renvoyer null.  
[AudioTrackList getTrackById\(id\)](#) [VideoTrackList getTrackById\(id\)](#) [AudioTrackVideoTrackAudioTrackListVideoTrackList](#)

Les objets [AudioTrack](#) et [VideoTrack](#) représentent des pistes spécifiques d'une [ressource multimédia](#) . Chaque piste peut avoir un identifiant, une catégorie, une étiquette et une langue. Ces aspects d'une piste sont permanents pendant toute la durée de vie de la piste ; même si une piste est supprimée d'une ressource [multimédia](#) [AudioTrackList](#) ou [VideoTrackList](#) d'objets, ces aspects ne changent pas.

De plus, [AudioTrack](#) les objets peuvent chacun être activés ou désactivés ; c'est l' *état activé* de la piste audio . Lorsqu'un [AudioTrack](#) est créé, son *état activé* doit être défini sur faux (désactivé). L' [algorithme d'extraction de ressources](#) peut remplacer cela.

De même, un seul [VideoTrack](#) objet par [VideoTrackList](#) objet peut être sélectionné, c'est l' *état de sélection* de la piste vidéo . Lorsqu'un [VideoTrack](#) est créé, son *état de sélection* doit être défini sur faux (non sélectionné). L' [algorithme d'extraction de ressources](#) peut remplacer cela.

Les attributs [AudioTrack id](#) et [VideoTrack id](#) doivent retourner l'identifiant de la piste, si elle en a un, ou la chaîne vide sinon. Si la [ressource multimédia](#) est dans un format qui prend en charge [la syntaxe de fragment multimédia](#) , l' identifiant retourné pour une piste particulière doit être le même identifiant qui activerait la piste s'il était utilisé comme nom de piste dans la dimension piste d' un tel [fragment](#) . [\[INBAND\]](#)

Par exemple, dans les fichiers Ogg, ce serait le champ d'en-tête Nom de la piste. [\[OGGSKELETONHEADERS\]](#)

Les attributs [AudioTrack kind](#) et [VideoTrack kind](#) doivent retourner la catégorie de la piste, si elle en a une, ou la chaîne vide sinon.

La catégorie d'une piste est la chaîne donnée dans la première colonne du tableau ci-dessous qui est la plus appropriée pour la piste en fonction des définitions dans les deuxième et troisième colonnes du tableau, telles que déterminées par les métadonnées incluses dans la piste dans la ressource [multimédia](#) . La cellule de la troisième colonne d'une ligne indique à quoi s'applique la catégorie donnée dans la cellule de la première colonne de cette ligne ; une catégorie n'est appropriée pour une piste audio que si elle s'applique aux pistes audio, et une catégorie n'est appropriée pour les pistes vidéo que si elle s'applique aux pistes vidéo. Les catégories ne doivent être renvoyées pour [AudioTrack](#) les objets que si elles sont appropriées pour l'audio, et ne doivent être renvoyées pour [VideoTrack](#) les objets que si elles sont appropriées pour la vidéo.

Pour les fichiers Ogg, le champ d'en-tête Rôle de la piste donne les métadonnées pertinentes. Pour les ressources multimédias DASH, l' `Role` élément véhicule l'information. Pour WebM, seul l' `FlagDefault` élément correspond actuellement à une valeur. *L'approvisionnement en pistes de ressources multimédias intrabande à partir de conteneurs multimédias vers HTML* contient plus de détails. [\[OGGSKELETONHEADERS\]](#) [\[DASH\]](#) [\[WEBMCG\]](#) [\[INBAND\]](#)

Valeurs de retour pour [AudioTrack's kind](#) et [VideoTrack's kind](#)

Catégorie	Définition	S'applique à...	Exemples
" <b>alternative</b> "	Une alternative possible à la piste principale, par exemple une prise	Audio et vidéo.	Ogg : "audio/alternatif" ou "vidéo/alternatif" ; DASH : "alternatif" sans les rôles "principal" et "commentaire", et,

Valeurs de retour pour [AudioTrack's kind](#) et [VideoTrack's kind](#)

Catégorie	Définition	S'applique à...	Exemples
	différente d'une chanson (audio), ou un angle différent (vidéo).		pour l'audio, sans le rôle "dub" (les autres rôles sont ignorés).
" <b>captions</b> "	Une version de la piste vidéo principale avec des sous-titres gravés. (Pour le contenu hérité ; le nouveau contenu utiliserait des pistes de texte.)	Vidéo uniquement.	DASH : rôles "légende" et "principal" ensemble (les autres rôles sont ignorés).
" <b>descriptions</b> "	Une description audio d'une piste vidéo.	Audio seulement.	Ogg : "audio/audiodesc".
" <b>main</b> "	La piste audio ou vidéo principale.	Audio et vidéo.	Ogg : "audio/principal" ou "vidéo/principal" ; WebM : l'élément "FlagDefault" est défini ; DASH : rôle "principal" sans les rôles "caption", "subtitle" et "dub" (les autres rôles sont ignorés).
" <b>main-desc</b> "	La piste audio principale, mélangée avec des descriptions audio.	Audio seulement.	Audio AC3 en MPEG-2 TS : bsmod=2 et full_svc=1.
" <b>sign</b> "	Une interprétation en langue des signes d'une piste audio.	Vidéo uniquement.	Ogg : "vidéo/signe".
" <b>subtitles</b> "	Une version de la piste vidéo principale avec des sous-titres gravés. (Pour le contenu hérité ; le nouveau contenu utiliserait des pistes de texte.)	Vidéo uniquement.	DASH : rôles "sous-titre" et "principal" ensemble (les autres rôles sont ignorés).
" <b>translation</b> "	Une version traduite de la piste audio principale.	Audio seulement.	Ogg : "audio/dub". DASH : rôles "dub" et "main" ensemble (les autres rôles sont ignorés).

Valeurs de retour pour [AudioTrack's kind](#) et [VideoTrack's kind](#)

Catégorie	Définition	S'applique à...	Exemples
" <a href="#">commentary</a> "	Commentaire sur la piste audio ou vidéo principale, par exemple le commentaire d'un réalisateur.	Audio et vidéo.	DASH : rôle "commentaire" sans rôle "principal" (les autres rôles sont ignorés).
" " (chaîne vide)	Aucun genre explicite, ou le genre donné par les métadonnées de la piste n'est pas reconnu par l'agent utilisateur.	Audio et vidéo.	

Les attributs [AudioTrack label](#) et [VideoTrack label](#) doivent retourner le label de la piste, si elle en a un, ou la chaîne vide sinon. [\[INBAND\]](#)

Les attributs [AudioTrack language](#) et [VideoTrack language](#) doivent renvoyer la balise de langue BCP 47 de la langue de la piste, si elle en possède une, ou la chaîne vide dans le cas contraire. Si l'agent utilisateur n'est pas en mesure d'exprimer cette langue en tant que balise de langue BCP 47 (par exemple parce que les informations de langue dans le format de la [ressource multimédia](#) sont une chaîne de forme libre sans interprétation définie), alors la méthode doit renvoyer le vide chaîne, comme si la piste n'avait pas de langue. [\[INBAND\]](#)

L' [AudioTrack enabled](#) attribut, à l'obtention, doit retourner true si la piste est actuellement activée, et false sinon. Au réglage, il faut activer la piste si la nouvelle valeur est vraie, et la désactiver sinon. (Si la piste n'est plus dans un [AudioTrackList](#) objet, alors la piste activée ou désactivée n'a aucun effet au-delà de la modification de la valeur de l'attribut sur l' [AudioTrack](#) objet.)

Chaque fois qu'une piste audio dans un [AudioTrackList](#) qui était désactivée est activée, et chaque fois qu'une qui était activée est désactivée, l'agent utilisateur doit [mettre en file d'attente une tâche d'élément multimédia](#) donnée à l' [élément multimédia](#) pour [déclencher un événement](#) nommé [change](#) sur l' [AudioTrackList](#) objet.

Une piste audio qui n'a pas de données pour une position particulière sur la [chronologie du média](#) , ou qui n'existe pas à cette position, doit être interprétée comme étant silencieuse à ce point de la chronologie.

L' [VideoTrackList selectedIndex](#) attribut doit renvoyer l'index de la piste actuellement sélectionnée, le cas échéant. Si l' [VideoTrackList](#) objet ne

représente actuellement aucune piste, ou si aucune des pistes n'est sélectionnée, il doit à la place renvoyer -1.

L' [VideoTrack selected](#) attribut, à l'obtention, doit retourner true si la piste est actuellement sélectionnée, et false sinon. Au réglage, il faut sélectionner la piste si la nouvelle valeur est vraie, et la désélectionner sinon. Si la piste est dans un [VideoTrackList](#), tous les autres [VideoTrack](#) objets de cette liste doivent être désélectionnés. (Si la piste n'est plus dans un [VideoTrackList](#) objet, alors la piste sélectionnée ou désélectionnée n'a aucun effet au-delà de la modification de la valeur de l'attribut sur l' [VideoTrack](#) objet.)

Chaque fois qu'une piste dans a [VideoTrackList](#) qui n'était pas sélectionnée précédemment est sélectionnée, et chaque fois que la piste sélectionnée dans a [VideoTrackList](#) est désélectionnée sans qu'une nouvelle piste soit sélectionnée à sa place, l'agent utilisateur doit [mettre en file d'attente une tâche d'élément multimédia](#) étant donné l' [élément multimédia](#) pour [déclencher un événement](#) nommé [change](#) à l' [VideoTrackList](#) objet. Cette [tâche](#) doit être [mise en file d'attente](#) avant la [tâche](#) qui déclenche l' [resize](#) événement, le cas échéant.

Une piste vidéo qui n'a pas de données pour une position particulière sur la [chronologie du média](#) doit être interprétée comme étant [noire transparente](#) à ce point de la chronologie, avec les mêmes dimensions que la dernière image avant cette position, ou, si la position est avant toutes les données pour cette piste, les mêmes dimensions que la première image de cette piste. Une piste qui n'existe pas du tout à la position actuelle doit être traitée comme si elle existait mais n'avait pas de données.

Par exemple, si une vidéo a une piste qui n'est introduite qu'après une heure de lecture, et que l'utilisateur sélectionne cette piste puis revient au début, l'agent utilisateur agira comme si cette piste avait commencé au début de la ressource multimédia . mais était simplement transparent jusqu'à une heure.

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par tous les objets implémentant les interfaces [AudioTrackList](#) et [VideoTrackList](#) :

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
<a href="#">onchange</a> ✓ <a href="#">MDN</a>	<a href="#">change</a>
<a href="#">onaddtrack</a> ✓ <a href="#">MDN</a>	<a href="#">addtrack</a>

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
<code>onremovetrack</code> ✓ MDN	<code>removetrack</code>

#### 4.8.11.10.2 Sélection de pistes audio et vidéo spécifiques de manière déclarative

Les attributs `audioTracks` et `videoTracks` permettent aux scripts de sélectionner la piste à lire, mais il est également possible de sélectionner des pistes spécifiques de manière déclarative, en spécifiant des pistes particulières dans le [fragment](#) de l' [URL](#) de la [ressource multimédia](#) . Le format du [fragment](#) dépend du [type MIME](#) de la [ressource multimédia](#) . [\[RFC2046\]](#) [\[URL\]](#)

Dans cet exemple, une vidéo qui utilise un format qui prend en charge [la syntaxe de fragment multimédia](#) est intégrée de telle sorte que les angles alternatifs étiquetés "Alternative" sont activés à la place de la piste vidéo par défaut.

```
<video src="myvideo#track=Alternative"></video>
```

#### 4.8.11.11 Pistes de texte chronométrées

##### 4.8.11.11.1 Modèle de piste de texte

Un [élément multimédia](#) peut avoir un groupe de **pistes de texte** associées , connu sous le nom de **liste de pistes de texte** de [l'élément multimédia](#) . Les [pistes de texte](#) sont triées comme suit :

1. Les [pistes de texte](#) correspondant aux `track` éléments enfants de l' [élément média](#) , dans [l'ordre de l'arborescence](#) .
2. Toutes [les pistes de texte](#) ajoutées à l'aide de la `addTextTrack()` méthode, dans l'ordre dans lequel elles ont été ajoutées, la plus ancienne en premier.
3. Toutes [les pistes de texte spécifiques à la ressource multimédia](#) ( [pistes de texte](#) correspondant aux données de la [ressource multimédia](#) ), dans l'ordre défini par la spécification de format de la [ressource multimédia](#) .

Une [piste de texte](#) se compose de :

##### Le type de piste de texte

Cela décide comment la piste est gérée par l'agent utilisateur. Le genre est représenté par une chaîne. Les chaînes possibles sont :

- `subtitles`
- `captions`



- [descriptions](#)
- [chapters](#)
- [metadata](#)

Le [type de piste](#) peut changer dynamiquement, dans le cas d'une [piste de texte](#) correspondant à un [track](#)élément.

### **Une marque**

Il s'agit d'une chaîne lisible par l'homme destinée à identifier la piste pour l'utilisateur.

Le [libellé d'une piste](#) peut changer dynamiquement, dans le cas d'une [piste de texte](#) correspondant à un [track](#)élément.

Lorsqu'une [étiquette de piste de texte](#) est une chaîne vide, l'agent utilisateur devrait générer automatiquement une étiquette appropriée à partir des autres propriétés de la piste de texte (par exemple, le type de piste de texte et la langue de la piste de texte) pour l'utiliser dans son interface utilisateur. Cette étiquette générée automatiquement n'est pas exposée dans l'API.

### **Un type d'envoi de piste de métadonnées intrabande**

Il s'agit d'une chaîne extraite de la [ressource multimédia](#) spécifiquement pour les pistes de métadonnées intrabande afin de permettre à ces pistes d'être distribuées à différents scripts dans le document.

Par exemple, une chaîne de télévision traditionnelle diffusée en streaming sur le Web et complétée par des fonctionnalités interactives spécifiques au Web peut inclure des pistes de texte avec des métadonnées pour le ciblage publicitaire, des données de jeux-questionnaires pendant les jeux télévisés, les états des joueurs pendant les jeux sportifs, des informations sur les recettes pendant les programmes alimentaires et ainsi de suite. Au démarrage et à la fin de chaque programme, de nouvelles pistes peuvent être ajoutées ou supprimées du flux, et à mesure que chacune est ajoutée, l'agent utilisateur peut les lier à des modules de script dédiés en utilisant la valeur de cet attribut.

À l'exception des pistes de texte de métadonnées intrabande, le [type de répartition des pistes de métadonnées intrabande](#) est la chaîne vide. La façon dont cette valeur est renseignée pour différents formats de média est décrite dans [les étapes pour exposer une piste de texte spécifique à la ressource média](#) .

### **Une langue**

Il s'agit d'une chaîne (une balise de langue BCP 47) représentant la langue des repères de la piste de texte. [\[BCP47\]](#)

La [langue d'une piste de texte](#) peut changer dynamiquement, dans le cas d'une [piste de texte](#) correspondant à un [track](#)élément.

### **Un état de préparation**

L'un des éléments suivants :



***Pas chargé***

Indique que les repères de la piste de texte n'ont pas été obtenus.

***Chargement***

Indique que la piste de texte est en cours de chargement et qu'aucune erreur fatale n'a été rencontrée jusqu'à présent. D'autres repères peuvent encore être ajoutés à la piste par l'analyseur.

***Chargé***

Indique que la piste de texte a été chargée sans erreur fatale.

***Échec du chargement***

Indique que la piste de texte était activée, mais lorsque l'agent utilisateur a tenté de l'obtenir, cela a échoué d'une manière ou d'une autre (par exemple, l' [URL](#) n'a pas pu être [analysée](#) , erreur réseau, format de piste de texte inconnu). Certains ou tous les indices sont probablement manquants et ne seront pas obtenus.

L' [état de préparation](#) d'une [piste de texte](#) change dynamiquement à mesure que la piste est obtenue.

***Un mode***

L'un des éléments suivants :

***Désactivé***

Indique que la piste de texte n'est pas active. Autre que dans le but d'exposer la piste dans le DOM, l'agent utilisateur ignore la piste de texte. Aucune cue n'est active, aucun événement n'est déclenché et l'agent utilisateur ne tentera pas d'obtenir les cues de la piste.

***Caché***

Indique que la piste de texte est active, mais que l'agent utilisateur n'affiche pas activement les signaux. Si aucune tentative n'a encore été faite pour obtenir les repères de la piste, l'agent utilisateur effectuera une telle tentative momentanément. L'agent utilisateur maintient une liste des signaux actifs et les événements sont déclenchés en conséquence.

***Montrant***

Indique que la piste de texte est active. Si aucune tentative n'a encore été faite pour obtenir les repères de la piste, l'agent utilisateur effectuera une telle tentative momentanément. L'agent utilisateur maintient une liste des signaux actifs et les événements sont déclenchés en conséquence. De plus, pour les pistes de texte dont le [type](#) est [subtitles](#) ou [captions](#), les repères sont superposés sur la vidéo selon le cas ; pour les pistes de texte dont le [type](#) est [descriptions](#), l'agent utilisateur met les repères à la disposition de l'utilisateur d'une manière non visuelle ; et pour les pistes de texte dont le [type](#) est [chapters](#), l'agent utilisateur met à la disposition de l'utilisateur un mécanisme par lequel l'utilisateur peut naviguer vers n'importe quel point de la [ressource multimédia](#) en sélectionnant un repère.

### **Une liste de zéro ou plusieurs indices**

Une liste d' [indices de piste de texte](#) , ainsi que **des règles de mise à jour du rendu de la piste de texte** . Par exemple, pour WebVTT, les [règles de mise à jour de l'affichage des pistes de texte WebVTT](#) . [\[WEBVTT\]](#)

La [liste des repères d'une piste de texte](#) peut changer dynamiquement, soit parce que la [piste de texte](#) n'a [pas encore été chargée](#) ou est toujours [en cours de chargement](#) , soit en raison d'une manipulation du DOM.

Chaque [piste de texte](#) a un [TextTrack](#) objet correspondant.

---

Chaque [élément multimédia](#) a une **liste de pistes de texte en attente** , qui doit initialement être vide, un indicateur **de blocage sur l'analyseur** , qui doit initialement être faux, et un indicateur **did-perform-automatic-track-selection** , qui doit également être initialement faux .

Lorsque l'agent utilisateur doit **remplir la liste des pistes de texte en attente** d'un [élément multimédia](#) , l'agent utilisateur doit ajouter à la [liste des pistes de texte en attente de](#) l'élément chaque [piste de texte dans la liste des pistes de texte](#) de l' élément dont [le mode de piste de texte](#) n'est pas [désactivé](#) et dont [l'état de préparation de la piste de texte](#) est [en cours de chargement](#) .

Chaque fois que le [track](#) nœud parent d'un élément change, l'agent utilisateur doit supprimer la [piste de texte](#) correspondante de toute [liste de pistes de texte en attente](#) dans laquelle il se trouve.

Chaque fois que [l'état de préparation de la piste de texte](#) d'une piste de texte passe à [Loaded](#) ou [Failed to load](#) , l'agent utilisateur doit la supprimer de toute [liste de pistes de texte en attente](#) dans laquelle elle se trouve.

Lorsqu'un [élément multimédia](#) est créé par un [analyseur HTML](#) ou [un analyseur XML](#) , l'agent utilisateur doit définir le drapeau [blocked-on-parser](#) de l'élément sur true. Lorsqu'un [élément multimédia](#) est retiré de la [pile d'éléments ouverts](#) d'un [analyseur HTML](#) ou [d'un analyseur XML](#) , l'agent utilisateur doit [respecter les préférences de l'utilisateur pour la sélection automatique des pistes de texte](#) , [remplir la liste des pistes de texte en attente](#) et définir le [blocage sur l'analyseur](#) de l'élément. drapeau à faux.

Les [pistes de texte](#) d'un [élément multimédia](#) sont **prêtes** lorsque la [liste des pistes de texte en attente de l' élément](#) est vide et que l' indicateur [de blocage sur l'analyseur](#) de l'élément est faux.

Chaque [élément multimédia](#) a un **indicateur de notification de changement de piste de texte en attente** , qui doit initialement être désactivé.

Chaque fois qu'une [piste de texte](#) qui se trouve dans la [liste des pistes de texte](#) d'un [élément multimédia](#) a sa valeur de changement [de mode de piste de texte](#) , l'agent utilisateur doit exécuter les étapes suivantes pour l' [élément multimédia](#) :

1. Si l' [indicateur de notification de modification de la piste de texte en attente de l' élément multimédia](#) est défini, retour.
2. Définissez l' [indicateur de notification de changement de piste de texte en attente](#) de [l'élément multimédia](#) .
3. [Mettez en file d'attente une tâche d'élément multimédia](#) en fonction de l' [élément multimédia](#) pour exécuter ces étapes :
  1. Désactivez l' [indicateur de notification de changement de piste de texte en attente](#) de [l'élément multimédia](#) .
  2. [Déclenche un événement](#) nommé [changes](#) sur l' objet de l'attribut de [l'élément multimédia](#) [.textTracksTextTrackList](#)
4. Si l' [indicateur d'affiche](#) de [l'élément multimédia](#) n'est pas défini, exécutez les [marches temporelles par](#) étapes.

La [source de tâche](#) pour les [tâches](#) répertoriées dans cette section est la [source de tâche de manipulation DOM](#) .

---

Un **repère de piste de texte** est l'unité de données sensibles au temps dans une [piste de texte](#) , correspondant par exemple pour les sous-titres et les légendes au texte qui apparaît à un moment donné et disparaît à un autre moment.

Chaque [cue de piste de texte](#) se compose de :

#### ***Un identifiant***

Une chaîne arbitraire.

#### ***Une heure de début***

Le temps, en secondes et fractions de seconde, qui décrit le début de la plage des [données multimédias](#) auxquelles s'applique le signal.

#### ***Un temps de fin***

Le temps, en secondes et en fractions de seconde, qui décrit la fin de la plage des [données multimédias](#) auxquelles s'applique le repère, ou Infini positif pour un [repère de piste de texte illimité](#) .

#### ***Un indicateur de pause à la sortie***

Un booléen indiquant si la lecture de la [ressource multimédia](#) doit s'arrêter lorsque la fin de la plage à laquelle le repère s'applique est atteinte.

### Quelques données supplémentaires spécifiques au format

Champs supplémentaires, selon les besoins du format, y compris les données réelles de la mémoire. Par exemple, WebVTT a une [direction d'écriture de repère de piste de texte](#) et ainsi de suite. [\[WEBVTT\]](#)

Un **repère de piste de texte illimité** est un repère de piste de texte avec une [heure de fin de repère de piste de texte](#) réglée sur Infini positif. Un [repère de piste de texte illimité](#) actif ne peut pas devenir inactif par l'augmentation monotone habituelle de la [position de lecture actuelle](#) pendant la lecture normale (par exemple, un repère de métadonnées pour un chapitre dans un événement en direct sans heure de fin annoncée.)

*L' [heure de début](#) et l'[heure de fin du repère de la piste de texte](#) peuvent être négatives. (La [position de lecture actuelle](#) ne peut cependant jamais être négative, donc les cues entièrement avant le temps zéro ne peuvent pas être actifs.)*

Chaque [signal de piste de texte](#) a un [TextTrackCue](#) objet correspondant (ou plus précisément, un objet qui hérite de [TextTrackCue](#) - par exemple, les signaux WebVTT utilisent l' [VTT Cue](#) interface). La représentation en mémoire d'un [signal de piste de texte](#) [TextTrackCue](#) peut être modifiée dynamiquement via cette API. [\[WEBVTT\]](#)

Un [repère de piste de texte](#) est associé à [des règles de mise à jour du rendu de la piste de texte](#), tel que défini par la spécification du type spécifique de [repère de piste de texte](#). Ces règles sont utilisées spécifiquement lorsque l'objet représentant le signal est ajouté à un [TextTrack](#) objet à l'aide de la [addCue\(\)](#) méthode.

De plus, chaque [signal de piste de texte](#) contient deux informations dynamiques :

#### Le drapeau actif

Cet indicateur doit être initialement désactivé. Le drapeau est utilisé pour s'assurer que les événements sont déclenchés de manière appropriée lorsque la cue devient active ou inactive, et pour s'assurer que les bonnes cues sont rendues.

L'agent utilisateur doit désactiver cet indicateur de manière synchrone chaque fois que l' [indication de la piste de texte est supprimée de la liste des indications de la piste de texte](#) de sa piste de texte ; chaque fois que la [piste de texte](#) elle-même est supprimée de [la liste des pistes de texte](#) de son [élément multimédia](#) ou que son [mode de piste de texte est](#) désactivé ; et chaque fois que l' élément [média](#) est remodifié en . Lorsque le drapeau est désactivé de cette manière pour un ou plusieurs repères dans [les pistes de texte](#) qui [s'affichaient](#) [readyState](#) [HAVE NOTHING](#) avant l'incident concerné, l'agent utilisateur doit, après avoir désactivé le drapeau pour toutes les cues concernées, appliquer les [règles de mise à jour du rendu de la piste de texte](#) de ces [pistes de texte](#). Par exemple, pour [les pistes de texte](#) basées sur

WebVTT, les [règles de mise à jour de l'affichage des pistes de texte WebVTT](#) . [\[WEBVTT\]](#)

### **L' état d'affichage**

Ceci est utilisé dans le cadre du modèle de rendu, pour garder les repères dans une position cohérente. Il doit initialement être vide. Chaque fois que le [drapeau actif de repère de piste de texte](#) est désactivé, l'agent utilisateur doit vider l' [état d'affichage du repère de piste de texte](#) .

Les [repères de piste de texte des pistes de texte](#) d'un [élément multimédia](#) sont classés les uns par rapport aux autres dans l' **ordre des repères de piste de texte** , qui est déterminé comme suit : groupez d'abord les [repères](#) par leur [piste de texte](#) , les groupes étant triés dans le même ordre que leurs [pistes de texte](#) apparaissent dans la [liste des pistes de texte](#) de l'[élément média](#) ; ensuite, au sein de chaque groupe, [les cues](#) doivent être triées par leur [heure de début](#) , la plus ancienne en premier ; ensuite, toutes [les cues](#) avec la même [heure de début](#) doivent être triées par leur [heure de fin](#) , le plus récent en premier ; et enfin, tous [les repères](#) avec [des heures de fin](#) identiques doivent être triés dans l'ordre dans lequel ils ont été ajoutés pour la dernière fois à leur [liste de pistes de texte respective des repères](#) , le plus ancien en premier (par exemple, pour les repères d'un fichier WebVTT, ce serait initialement l'ordre dans lequel les repères figuraient dans le dossier). [\[WEBVTT\]](#)

#### **4.8.11.11.2 Recherche de pistes de texte dans la bande**

Une **piste de texte spécifique à une ressource multimédia** est une [piste de texte](#) qui correspond aux données trouvées dans la [ressource multimédia](#) .

Les règles de traitement et de rendu de ces données sont définies par les spécifications pertinentes, par exemple la spécification du format vidéo si la [ressource multimédia](#) est une vidéo. Vous trouverez des détails sur certains formats hérités dans *Sourcing In-band Media Resource Tracks from Media Containers into HTML* . [\[INBAND\]](#)

Lorsqu'une [ressource multimédia](#) contient des données que l'agent utilisateur reconnaît et prend en charge comme étant équivalentes à une [piste de texte](#) , l'agent utilisateur [exécute](#) les **étapes pour exposer une piste de texte spécifique à la ressource multimédia** avec les données pertinentes, comme suit.

1. Associez les données pertinentes à une nouvelle [piste de texte](#) et son nouvel [TextTrack](#) objet correspondant. La [piste de texte](#) est une [piste de texte spécifique à une ressource multimédia](#) .
2. Définissez le [type](#) , l'[étiquette](#) et la [langue](#) de la nouvelle [piste de texte](#) en fonction de la sémantique des données pertinentes, comme défini par la spécification pertinente. S'il n'y a pas d'étiquette dans ces données, l' [étiquette](#) doit être définie sur la chaîne vide.

3. Associez la [liste des repères de la piste de texte](#) aux [règles de mise à jour du rendu de la piste de texte](#) appropriée pour le format en question.
4. Si le [type](#) de la nouvelle piste de [texte](#) est ou , définissez le [type de répartition de la piste de métadonnées intrabande de la piste de texte](#) comme suit, en fonction du type de [ressource multimédia](#) : [chaptersmetadata](#)

**Si la [ressource média](#) est un fichier Ogg**

- Le [type de répartition de la piste de métadonnées intrabande de la piste de texte](#) doit être défini sur la valeur du champ d'en-tête Nom. [\[OGGSKELETONHEADERS\]](#)

**Si la [ressource multimédia](#) est un fichier WebM**

- Le [type de répartition de la piste de métadonnées intrabande de la piste de texte](#) doit être défini sur la valeur de l' `CodecID` élément. [\[WEBMCG\]](#)

**Si la [ressource multimédia](#) est un fichier MPEG-2**

Soit *le type de flux* la valeur du champ "stream\_type" décrivant le type de piste de texte dans la section de carte de programme du fichier, interprété comme un entier non signé de 8 bits. Soit *longueur* la valeur du champ "ES\_info\_length" pour la piste dans la même partie de la section de plan de programme, interprétée comme un entier tel que défini par *Codage générique des images animées et des informations audio associées* . Soit *les octets du descripteur* les octets de *longueur* suivant le champ "ES\_info\_length".octets, exprimés en hexadécimal à l'aide [de chiffres hexadécimaux supérieurs ASCII](#) . [\[MPEG2\]](#)

**Si la [ressource multimédia](#) est un fichier MPEG-4**

Soit la première `stsd`case de la première `stbl`case de la première `minf`case de la première `mdia`case de la case de la piste de [texte](#)`trak` dans la première `moov`case du fichier soit la *case stsd* , s'il y en a une. Si le fichier n'a pas de *boîte stsd* , ou si la *boîte stsd* n'a ni `mett`boîte ni `metx`boîte, le [type d'envoi de la piste de métadonnées intrabande de la piste de texte](#) doit être défini sur la chaîne vide. Sinon, si la *boîte stsd* a une `mett`boîte, le [type de répartition de la piste de métadonnées intrabande de la piste de texte](#) doit être défini sur la concaténation de la chaîne "mett", un caractère U+0020 SPACE, et la valeur du premier `mime_format`champ de la première `mett`case de la *stsd box* , ou la chaîne vide si ce champ est absent dans cette case. Sinon, si la *stsd box* n'a pas `mett`de case mais a un `metx`box alors le [type de répartition de la piste de métadonnées intrabande de la piste de texte](#) doit être défini sur la concaténation de la chaîne " metx", d'un caractère U+0020 SPACE et de la valeur du premier `namespace`champ de la première `metx`case de la *stsd box* , ou du vide chaîne si ce champ est absent dans cette boîte. [\[MPEG4\]](#)

5. Remplissez la [liste des signaux de](#) la nouvelle [piste de texte](#) avec les signaux analysés jusqu'à présent, en suivant les [directives d'exposition des signaux](#) , et commencez à la mettre à jour dynamiquement si nécessaire.
6. Définissez l' [état de préparation](#) de la nouvelle [piste de texte](#) sur [chargé](#) .



7. Définissez le [mode](#) de la nouvelle piste de [texte](#) sur le mode compatible avec les préférences de l'utilisateur et les exigences de la spécification pertinente pour les données.

*Par exemple, s'il n'y a pas d'autres sous-titres actifs et qu'il s'agit d'une piste de sous-titres forcés (une piste de sous-titres donnant des sous-titres dans la langue principale de la piste audio, mais uniquement pour l'audio qui est en fait dans une autre langue), alors ces sous-titres peuvent être activés ici .*

8. Ajoutez la nouvelle [piste de texte](#) à la [liste des pistes de texte](#) de [l'élément multimédia](#) .
9. [Déclenchez un événement](#) nommé [addtrack](#) sur l' objet de l'attribut de [l'élément multimédia](#) , en utilisant , avec l' attribut initialisé sur l' objet de la [piste de texte](#) .[textTracksTextTrackListTrackEventtrackTextTrack](#)

#### 4.8.11.11.3 Recherche de pistes de texte hors bande

Lorsqu'un [track](#)élément est créé, il doit être associé à une nouvelle [piste de texte](#) (avec sa valeur définie comme définie ci-dessous) et son nouvel [TextTrack](#)objet correspondant.

Le [type de piste de texte](#) est déterminé à partir de l'état de l' [kind](#)attribut de l'élément selon le tableau suivant ; pour un état donné dans une cellule de la première colonne, le [kind](#) est la chaîne donnée dans la deuxième colonne :

État	Chaîne
<a href="#">Les sous-titres</a>	<a href="#">subtitles</a>
<a href="#">Légendes</a>	<a href="#">captions</a>
<a href="#">Descriptions</a>	<a href="#">descriptions</a>
<a href="#">Métadonnées des chapitres</a>	<a href="#">chapters</a>
<a href="#">Métadonnées</a>	<a href="#">metadata</a>

L' [étiquette de piste de texte](#) est l' [étiquette de piste](#) de l'élément .

La [langue de la piste de texte](#) est la [langue de la piste](#) de l'élément , le cas échéant, ou la chaîne vide dans le cas contraire.

Lorsque les attributs [kind](#), [label](#)et [srclang](#)sont définis, modifiés ou supprimés, la [piste de texte](#) doit être mise à jour en conséquence, conformément aux définitions ci-dessus.

*Les modifications apportées à l' [URL de la piste](#) sont gérées dans l'algorithme ci-dessous.*

L' [état de préparation de la piste de texte](#) n'est initialement [pas chargé](#) et le [mode de piste de texte](#) est initialement [désactivé](#) .

La [liste des pistes de texte des repères](#) est initialement vide. Il est modifié dynamiquement lorsque le fichier référencé est analysé. A la liste sont associées les [règles de mise à jour du rendu de la piste de texte](#) adaptée au format considéré ; pour WebVTT, il s'agit des [règles de mise à jour de l'affichage des pistes de texte WebVTT](#) . [\[WEBVTT\]](#)

Lorsque l' [track](#)élément parent d'un élément change et que le nouveau parent est un [élément média](#) , l'agent utilisateur doit alors ajouter la [piste de texte](#)[track](#) correspondante de l'élément à la [liste des pistes de texte](#) de l' [élément média](#) , puis [mettre en file d'attente une tâche d'élément média](#) en fonction de l' [élément média](#) à [déclencher un événement](#) nommé à l' objet de l'attribut de [l'élément multimédia](#) , en utilisant , avec l' attribut initialisé à l' objet de la [piste de texte](#) [.addtracktextTracksTextTrackListTrackEventtrackTextTrack](#)

Lorsque l' [track](#)élément parent d'un élément change et que l'ancien parent était un [élément média](#) , l'agent utilisateur doit supprimer la [piste de texte](#)[track](#) correspondante de l'élément de la [liste des pistes de texte de l'élément média](#) , puis [mettre en file d'attente une tâche d'élément média](#) en fonction de l' [élément média](#) à [déclencher un événement](#) nommé à l' objet de l'attribut de [l'élément multimédia](#) , en utilisant , avec l' attribut initialisé à l' objet de la [piste de texte](#) [.removetracktextTracksTextTrackListTrackEventtrackTextTrack](#)

---

Lorsqu'une [piste de texte](#) correspondant à un [track](#)élément est ajoutée à la [liste des pistes de texte](#) d' [un élément média](#) , l' agent utilisateur doit [mettre en file d'attente une tâche d' élément média](#) en fonction de l' [élément média](#) pour exécuter les étapes suivantes pour l' [élément média](#) :

1. [Si l'indicateur de blocage sur l'analyseur](#) de l'élément est vrai, alors retournez.
2. Si l'indicateur [did-perform-automatic-track-selection](#) de l'élément est vrai, alors retournez.
3. [Respectez les préférences de l'utilisateur pour la sélection automatique de la piste de texte](#) pour cet élément.

Lorsque l'agent utilisateur doit **respecter les préférences de l'utilisateur pour la sélection automatique de piste de texte** pour un [élément multimédia](#) , l'agent utilisateur doit exécuter les étapes suivantes :



1. [Effectuez une sélection automatique de piste de texte](#) pour [subtitles](#) et [captions](#).
2. [Effectuez une sélection automatique de piste de texte](#) pour [descriptions](#).
3. S'il existe des [pistes de texte](#) dans la [liste des pistes de texte](#) de l' [élément multimédia](#) dont [le type de piste de texte](#) est ou qui correspondent à des éléments avec un ensemble d'attributs dont [le mode de piste de texte](#) est désactivé , réglez le [mode de piste de texte](#) de toutes ces pistes sur [cachéchaptersmetadatatrackdefault](#)
4. Définissez l' indicateur [did-perform-automatic-track-selection](#) de l'élément sur true.

Lorsque les étapes ci-dessus indiquent d' **effectuer une sélection automatique de piste de texte** pour un ou plusieurs [types de piste de texte](#) , cela signifie exécuter les étapes suivantes :

1. Soit *les candidats* une liste composée des [pistes de texte](#) dans la [liste des pistes de texte](#) de l' [élément média](#) dont [le type de piste de texte](#) est l'un des types qui ont été transmis à l'algorithme, le cas échéant, dans l'ordre indiqué dans la [liste des pistes de texte](#) .
2. Si *les candidats* sont vides, alors retournez.
3. Si l'une des [pistes de texte](#) des *candidats* a un [mode de piste de texte](#) défini sur [Affichage](#) , revenez.
4. Si l'utilisateur a exprimé le souhait d'activer une piste parmi *les candidats* en fonction du [type de piste de texte](#) , [de la langue de la piste de texte](#) et [de l'étiquette de la piste de texte](#) , définissez son [mode de piste de texte](#) sur [Affichage](#) .

*Par exemple, l'utilisateur peut avoir défini une préférence de navigateur sur l'effet "Je veux des sous-titres en français chaque fois que possible", ou "S'il y a une piste de sous-titres avec 'Commentaire' dans le titre, activez-la", ou "S'il y a de l'audio pistes de description disponibles, activez-en une, idéalement en suisse allemand, mais à défaut en suisse allemand standard ou en allemand standard".*

Sinon, s'il existe des [pistes de texte](#) dans *les candidats* qui correspondent à [track](#) des éléments avec un [default](#) ensemble d'attributs dont [le mode de piste de texte](#) est défini sur [disabled](#) , définissez le [mode de piste de texte](#) de la première de ces pistes sur [display](#) .

Lorsqu'une [piste de texte](#) correspondant à un [track](#) élément rencontre l'une des circonstances suivantes, l'agent utilisateur doit [démarrer le track modèle de traitement](#) pour cette [piste de texte](#) et son [track](#) élément :

- L' [track](#) élément est créé.

- La [piste de texte](#) voit son [mode de piste de texte](#) modifié.
- L' [track](#)élément parent de l'élément change et le nouveau parent est un [élément multimédia](#) .

Lorsqu'un agent utilisateur doit **démarrer le [track](#)modèle de traitement** pour une [piste de texte](#) et son [track](#)élément, il doit exécuter l'algorithme suivant. Cet algorithme interagit étroitement avec le mécanisme [de boucle d'événements](#) ; en particulier, il a une [section synchrone](#) (qui est déclenchée dans le cadre de l'algorithme [de boucle d'événement](#) ). Les étapes de cette section sont marquées d'un ⌚.

1. Si une autre occurrence de cet algorithme est déjà en cours d'exécution pour cette [piste de texte](#) et son [track](#)élément, retournez, laissant cet autre algorithme s'occuper de cet élément.
2. Si le [mode de suivi de texte](#) de la piste de texte n'est pas défini sur [masqué](#) ou [affiché](#) , revenez.
3. Si l' élément de [la piste de texte](#)[track](#) n'a pas d' [élément média](#) comme parent, retournez.
4. Exécutez le reste de ces étapes [en parallèle](#) , en permettant à tout ce qui a provoqué l'exécution de ces étapes de continuer.
5. *En haut* : [Attendez un état stable](#) . La [section synchrone](#) comprend les étapes suivantes. (Les étapes de la [section synchrone](#) sont marquées d'un ⌚.)
6. ⌚ Définissez l' [état de préparation de la piste de texte](#) sur [chargement](#) .
7. ⌚ Soit *URL* l' [URL de piste](#) de l' [track](#)élément.
8. ⌚ Si le [track](#)parent de l'élément est un [élément multimédia](#) , laissez *corsAttributeState* être l'état de l' attribut content de l' [élément multimédia](#)[crossorigin](#) parent . Sinon, laissez *corsAttributeState* être [No CORS](#) .
9. Terminez la [section synchrone](#) en poursuivant les étapes restantes [en parallèle](#) .
10. Si l'*URL* n'est pas la chaîne vide, alors :
  1. Soit *request* le résultat de [la création d'une requête CORS potentielle](#) donnée *URL* , " [track](#)" et *corsAttributeState* , et avec l' *indicateur de secours de même origine* défini.
  2. Définissez [le client](#) de *la demande* sur l' [objet de paramètres pertinent](#) du [document de nœud](#) de l'élément [.track](#)
  3. Définissez [le type](#) d'initiateur de *la requête* sur " [.track](#)

#### 4. [Récupérer la requête](#) .

Les [tâches mises en file d'](#) attente par l'algorithme d'extraction sur la [source de tâches réseau](#) pour traiter les données pendant leur extraction doivent déterminer le type de la ressource. Si le type de ressource n'est pas un format de piste de texte pris en charge, le chargement échouera, comme décrit ci-dessous. Sinon, les données de la ressource doivent être transmises à l'analyseur approprié (par exemple, l' [analyseur WebVTT](#) ) au fur et à mesure de leur réception, la [liste des pistes de texte des repères](#) étant utilisée pour la sortie de cet analyseur. [\[WEBVTT\]](#)

*L'analyseur approprié mettra à jour de manière incrémentielle la [liste des pistes de texte des repères](#) pendant ces [tâches de source de tâche réseau](#) , car chacune de ces tâches est exécutée avec les données reçues du réseau).*

Cette spécification ne dit pas actuellement si ou comment vérifier les types MIME des pistes de texte, ou si ou comment effectuer un reniflage de type de fichier en utilisant les données de fichier réelles. Les responsables de la mise en œuvre diffèrent dans leurs intentions à ce sujet et il est donc difficile de savoir quelle est la bonne solution. En l'absence de toute exigence ici, l'exigence stricte des spécifications HTTP de suivre l'en-tête Content-Type prévaut ("Content-Type spécifie le type de média des données sous-jacentes." ... "Si et seulement si le type de média n'est pas donné par un champ Content-Type, le destinataire PEUT tenter de deviner le type de support via l'inspection de son contenu et/ou de l'extension ou des extensions de nom de l'URI utilisé pour identifier la ressource").

Si la récupération échoue pour une raison quelconque (erreur réseau, le serveur renvoie un code d'erreur, CORS échoue, etc.), ou si l'URL est la chaîne vide, mettez [en file d'attente une tâche d'élément](#) sur la [source de la tâche de manipulation DOM](#) en fonction de l' [élément multimédia](#) à modifier en premier l' [état de préparation de la piste de texte](#) n'a [pas pu être chargé](#) , puis [déclenche un événement](#) nommé [error](#) au niveau de l' [track](#) élément.

Si la récupération n'échoue pas, mais que le type de ressource n'est pas un format de piste de texte pris en charge, ou que le fichier n'a pas été traité avec succès (par exemple, le format en question est un format XML et le fichier contient une erreur de bonne formation requise par XML être détectée et signalée à l'application), la [tâche](#) qui est [mise en file d'attente](#) sur la [source de tâches réseau](#) dans laquelle le problème susmentionné est détecté doit changer l' [état de préparation de la piste de texte](#) en [échec de chargement](#) et [de déclenchement d'un événement](#) nommé [error](#) au niveau de l' [track](#) élément.

Si la récupération n'échoue pas et que le fichier a été traité avec succès, la [tâche](#) finale qui est [mise en file d'attente](#) par la [source de la tâche réseau](#) , après avoir terminé l'analyse des données, doit changer l' [état de préparation de la piste de texte](#) en [chargé](#) et [déclencher un événement](#) nommé [load](#) à l' [track](#) élément.

Si, pendant que la récupération est en cours, soit :

5. l' [URL de la piste](#) change de sorte qu'elle n'est plus égale à *URL* , tandis que le [mode de piste de texte](#) est défini sur [masqué](#) ou [affiché](#) ; ou
6. le [mode de suivi du texte](#) passe à [masqué](#) ou [affiché](#) , tandis que l' [URL de la piste](#) n'est pas égale à l'*URL*

... alors l'agent utilisateur doit abandonner [la récupération](#) , en supprimant toutes [les tâches](#) en attente générées par cet algorithme (et en particulier, en n'ajoutant aucun signal à la [liste des signaux de la piste de texte](#) après le moment où l'URL a changé), puis [mettre en file d'attente une tâche d'élément](#) sur la [source de la tâche de manipulation DOM](#) étant donné l' [track](#)élément qui modifie d'abord l' [état de préparation de la piste de texte](#) en [échec de chargement](#) , puis [déclenche un événement](#) nommé [error](#) sur l' [track](#)élément.

11. Attendez que l' [état de préparation de la piste de texte](#) ne soit plus défini sur [chargement](#) .
12. Attendez que l' [URL de la piste](#) ne soit plus égale à *URL* , en même temps que le [mode de piste de texte](#) est défini sur [masqué](#) ou [affiché](#) .
13. Passez à l'étape intitulée *top* .

Chaque fois qu'un [track](#)élément a son [src](#)attribut défini, modifié ou supprimé, l'agent utilisateur doit [immédiatement](#) vider la [liste des repères](#) de la [piste de](#) texte de l'élément . (Cela entraîne également l'algorithme ci-dessus pour arrêter d'ajouter des indices à partir de la ressource obtenue à l'aide de l'URL précédemment donnée, le cas échéant.)

#### **4.8.11.11.4 Directives pour exposer des repères dans divers formats en tant que [repères de piste de texte](#)**

La façon dont les repères de piste de texte d'un format spécifique doivent être interprétés pour les besoins du traitement par un agent utilisateur HTML est définie par ce format. En l'absence d'une telle spécification, cette section fournit certaines contraintes dans lesquelles les implémentations peuvent tenter d'exposer de manière cohérente de tels formats.

Pour prendre en charge le modèle [de piste de texte](#) HTML, chaque unité de données chronométrées est convertie en un [repère de piste de texte](#) . Lorsque le mappage des caractéristiques du format aux aspects d'un [signal de piste de texte](#) tel que défini dans la présente spécification n'est pas défini, les implémentations doivent s'assurer que le mappage est cohérent avec les définitions des aspects d'un [signal de piste de texte](#) tels que définis ci-dessus, ainsi que avec les contraintes suivantes :

## L' identificateur de repère de la piste de texte

Doit être défini sur la chaîne vide si le format n'a pas d'analogue évident avec un identifiant par cue.

## L' indicateur de pause à la sortie de la piste de texte

Doit être défini sur faux.

### 4.8.11.11.5 API de suivi de texte



[Exposed=Window]

```
interface TextTrackList : EventTarget {
```

```
  readonly attribute unsigned long length;
```

```
  getter TextTrack (unsigned long index);
```

```
  TextTrack? getTrackById(DOMString id);
```

```
  attribute EventHandler onchange;
```

```
  attribute EventHandler onaddtrack;
```

```
  attribute EventHandler onremovetrack;
```

```
};
```

**`media.textTracks.length`**



Renvoie le nombre de [pistes de texte](#) associées à l'[élément média](#) (par exemple à partir [track](#)d'éléments). Il s'agit du nombre de [pistes de texte](#) dans la [liste des pistes de texte](#) de l'[élément multimédia](#) .

**`media.textTracks[ n ]`**

Renvoie l' [TextTrack](#) objet représentant la *n* ième [piste de texte](#) dans la [liste des pistes de texte](#) de l'[élément multimédia](#) .

**`textTrack = media.textTracks.getTrackById(id)`**



Renvoie l' [TextTrack](#) objet avec l'identifiant donné, ou null si aucune piste n'a cet identifiant.

Un [TextTrackList](#) objet représente une liste mise à jour dynamiquement de [pistes de texte](#) dans un ordre donné.

L' `textTracks` attribut des [éléments média](#) doit retourner un [TextTrackList](#) objet représentant les [TextTrack](#) objets des [pistes de texte](#) dans la [liste des pistes de texte](#) de l' [élément média](#) , dans le même ordre que dans la [liste des pistes de texte](#) .



L' `length` attribut d'un [TextTrackList](#) objet doit renvoyer le nombre de [pistes de texte](#) dans la liste représentée par l' [TextTrackList](#) objet.

Les [indices de propriété pris en charge](#) d'un [TextTrackList](#) objet à tout instant sont les nombres de zéro au nombre de [pistes de texte](#) dans la liste représentée par l' [TextTrackList](#) objet moins un, le cas échéant. S'il n'y a pas [de pistes de texte](#) dans la liste, il n'y a pas [d'index de propriété pris en charge](#) .

Pour [déterminer la valeur d'une propriété indexée](#) d'un [TextTrackList](#) objet pour un index `index` donné , l'agent utilisateur doit retourner l' `index` ième [piste de texte](#) dans la liste représentée par l' [TextTrackList](#) objet.

La méthode doit renvoyer le premier dans l' objet dont l'attribut IDL renverrait une valeur égale à la valeur de l' argument `id` . Lorsqu'aucune piste ne correspond à l'argument donné, la méthode doit renvoyer `null`. [getTrackById\(id\) TextTrackTextTrackListid](#)



```
enum TextTrackMode { "disabled", "hidden", "showing" };
```

```
enum TextTrackKind { "subtitles", "captions", "descriptions",  
"chapters", "metadata" };
```

```
[Exposed=Window]
```

```
interface TextTrack : EventTarget {
```

```
  readonly attribute TextTrackKind kind;
```

```
readonly attribute DOMString label;
```

```
readonly attribute DOMString language;
```

```
readonly attribute DOMString id;
```

```
readonly attribute DOMString inBandMetadataTrackDispatchType;
```

```
attribute TextTrackMode mode;
```

```
readonly attribute TextTrackCueList? cues;
```

```
readonly attribute TextTrackCueList? activeCues;
```

```
undefined addCue (TextTrackCue cue);
```

```
undefined removeCue (TextTrackCue cue);
```

```
attribute EventHandler oncuechange;
```

```
};
```

```
textTrack = media.addTextTrack(kind [, label [, language ] ])
```

Crée et renvoie un nouvel [TextTrack](#) objet, qui est également ajouté à la [liste des pistes de texte](#) de [l'élément multimédia](#) .

```
textTrack.kind
```

✓ 

Renvoie la chaîne [de type de la piste de texte](#) .

```
textTrack.label
```

✓ 

Renvoie l' [étiquette de piste de texte](#) , s'il y en a une, ou la chaîne vide sinon (indiquant qu'une étiquette personnalisée doit probablement être générée à partir des autres attributs de l'objet si l'objet est exposé à l'utilisateur).

```
textTrack.language
```

✓ 

Renvoie la chaîne [de langue de la piste de texte](#) .

`textTrack.id`

✓

Renvoie l'ID de la piste donnée.

Pour les pistes intrabande, il s'agit de l'ID qui peut être utilisé avec un [fragment](#) si le format prend en charge [la syntaxe de fragment de média](#) et qui peut être utilisé avec la [getTrackById\(\)](#) méthode.

Pour [TextTrack](#) les objets correspondant à [track](#) des éléments, il s'agit de l'ID de l' [track](#) élément.

`textTrack.inBandMetadataTrackDispatchType`

✓

Renvoie la chaîne [de type de répartition de la piste de métadonnées intrabande de la piste de texte](#) .

`textTrack.mode [ = value ]`

✓

Renvoie le [mode de suivi du texte](#) , représenté par une chaîne de la liste suivante :

" [disabled](#) "

Le mode [piste de texte désactivé](#) .

" [hidden](#) "

Le mode [caché de la piste de texte](#) .

" [showing](#) "

Le mode [d'affichage de la piste de texte](#) .

Peut être réglé, pour changer le mode.

`textTrack.cues`

✓

Renvoie la [liste des pistes de texte des repères](#) , sous la forme d'un [TextTrackCueList](#) objet.

`textTrack.activeCues`

✓

Renvoie les [cues de la piste de texte](#) de la [liste des pistes de texte des cues](#) qui sont actuellement actives (c'est-à-dire qui commencent avant la [position de lecture actuelle](#) et se terminent après), sous forme d' [TextTrackCueList](#) objet.

`textTrack.addCue (cue)`

✓

Ajoute la réplique donnée à [la liste des pistes de texte](#) de *textTrack* .

`textTrack.removeCue (cue)`





Supprime la réplique donnée de [la liste des pistes de texte](#) de *textTrack* .

La méthode des [éléments multimédias](#) , lorsqu'elle est invoquée, doit exécuter les étapes suivantes :`addTextTrack(kind, label, language)`

1. Créez un nouvel `TextTrack` objet.
2. Créez une nouvelle [piste de texte](#) correspondant au nouvel objet et définissez son [type de piste de texte](#) sur *kind* , son étiquette [de piste de texte](#) sur *label* , sa [langue de piste de texte](#) sur *language* , son [état de préparation de la piste de texte](#) sur l' état [chargé de la piste de texte](#) , son [mode de piste de texte](#) au mode [caché de piste de texte](#) , et sa [liste de pistes de texte de repères](#) à une liste vide.

Initialement, la [liste des repères de la piste de texte](#) n'est associée à aucune [règle de mise à jour du rendu de la piste de texte](#) . Lorsqu'une [cue de piste de texte](#) y est ajoutée, la [liste des cues de la piste de texte](#) a ses règles définies en permanence en conséquence.

3. Ajoutez la nouvelle [piste de texte](#) à la [liste des pistes de texte](#) de l'[élément multimédia](#) .
4. [Mettez en file d'attente une tâche d'élément multimédia](#) donnée à l' [élément multimédia](#) pour [déclencher un événement](#) nommé `addtrack` sur l' objet de l'attribut de [l'élément multimédia](#) , en utilisant , avec l' attribut initialisé sur l'objet de la nouvelle piste de [texte](#) .`textTracksTextTrackListTrackEventtrackTextTrack`
5. Retourne le nouvel `TextTrack` objet.

---

L' `kind` attribut doit renvoyer le [type de piste de texte](#) de la [piste de texte](#) représentée par l' `TextTrack` objet.

L' `label` attribut doit renvoyer l' [étiquette de piste de texte](#) de la [piste de texte](#) représentée par l' `TextTrack` objet.

L' `language` attribut doit renvoyer la [langue de la piste de texte](#) de la [piste de texte](#) représentée par l' `TextTrack` objet.

L' `id` attribut renvoie l'identifiant de la piste, s'il en a un, ou la chaîne vide dans le cas contraire. Pour les pistes qui correspondent à `track` des éléments, l'identifiant de la piste est la valeur de l' `id` attribut de l'élément, le cas échéant. Pour les pistes intrabande, l'identifiant de la piste est spécifié par la [ressource multimédia](#) . Si

la [ressource multimédia](#) est dans un format qui prend en charge [la syntaxe de fragment multimédia](#) , l' identifiant retourné pour une piste particulière doit être le même identifiant qui activerait la piste s'il était utilisé comme nom de piste dans la dimension piste d' un tel [fragment](#) .

L' [inBandMetadataTrackDispatchType](#) attribut doit renvoyer le [type de distribution de piste de métadonnées intrabande](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente.

L' [mode](#) attribut, lors de l'obtention, doit renvoyer la chaîne correspondant au [mode de suivi de texte](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente, tel que défini par la liste suivante :

" [disabled](#) "

Le mode [piste de texte désactivé](#) .

" [hidden](#) "

Le mode [caché de la piste de texte](#) .

" [showing](#) "

Le mode [d'affichage de la piste de texte](#) .

Lors de la définition, si la nouvelle valeur n'est pas égale à ce que l'attribut renverrait actuellement, la nouvelle valeur doit être traitée comme suit :

**Si la nouvelle valeur est " [disabled](#) "**

Définissez le [mode de piste de texte](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente sur le mode [de piste de texte désactivé](#) .

**Si la nouvelle valeur est " [hidden](#) "**

Définissez le [mode de piste de texte](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente sur le mode [masqué de la piste de texte](#) .

**Si la nouvelle valeur est " [showing](#) "**

Définissez le [mode de piste de texte](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente sur le mode [d'affichage de la piste de texte](#) .

Si le [mode de piste de texte](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente n'est pas le mode [de piste de texte désactivé](#) [cues](#) , alors l' attribut doit renvoyer un objet [actif](#) [TextTrackCueList](#) qui représente le sous-ensemble de la [liste de piste de texte des repères](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente dont la [fin les heures](#) se produisent à ou après la [première position possible lorsque le script a commencé](#) , dans [l'ordre de repère de la piste de texte](#) . Sinon, il doit retourner null. Pour chaque [TextTrack](#) objet, lorsqu'un objet est retourné, le même [TextTrackCueList](#) objet doit être retourné à chaque fois.

La [première position possible au démarrage du script](#) est la [première position possible](#) la dernière fois que la [boucle d'événements](#) a atteint l'étape 1.

Si le [mode de piste de texte](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente n'est pas le mode [de piste de texte désactivé](#) [activeCues](#) , alors l' attribut doit renvoyer un objet [actif](#) [TextTrackCueList](#) qui représente le sous-ensemble de la [liste de piste de texte des repères](#) de la [piste de texte](#) que l' [TextTrack](#) objet représente dont le [mode actif](#) [L'indicateur a été défini au démarrage du script](#) , dans [l'ordre de repère de la piste de texte](#) . Sinon, il doit retourner null. Pour chaque [TextTrack](#) objet, lorsqu'un objet est retourné, le même [TextTrackCueList](#) objet doit être retourné à chaque fois.

L' indicateur actif d' un [signal de piste de texte](#) [a été défini au démarrage du script](#) si son [indicateur actif de signal de piste de texte](#) a été défini la dernière fois que la [boucle d' événement](#) a atteint [l' étape 1](#) .

---

La méthode des objets, lorsqu'elle est invoquée, doit exécuter les étapes suivantes : [addCue \(cue\)](#) [TextTrack](#)

1. Si la [liste des pistes de texte des repères](#) n'a pas encore de [règles associées pour mettre à jour le rendu de la piste de texte](#) , alors associez la [liste des pistes de texte des repères](#) aux [règles de mise à jour du rendu de la piste de texte](#) appropriée au repère .
2. Si [la liste des pistes de texte des](#) règles associées aux cues [pour mettre à jour le rendu de la piste de texte](#) ne sont pas les mêmes [règles pour mettre à jour le rendu de la piste de texte](#) comme approprié pour *la cue* , lancez alors un [" InvalidStateError" DOMException](#) .
3. Si la *cue* donnée est dans une [liste de pistes de texte de cues](#) , alors supprimez *la cue* de cette [liste de pistes de texte de cues](#) .
4. Ajouter *un repère* à la [liste des repères](#) de [la piste de](#) [TextTrack](#) texte de l'objet .

La méthode des objets, lorsqu'elle est invoquée, doit exécuter les étapes suivantes : [removeCue \(cue\)](#) [TextTrack](#)

1. Si le *signal* donné n'est pas dans la [liste des signaux de la piste de](#) [texte](#) [TextTrack](#) de l'objet , lancez un [" " .NotFoundError DOMException](#)
2. Supprimer *le repère* de la [liste des repères](#) de [la piste de texte](#) [TextTrack](#) de l'objet .

Dans cet exemple, un [audio](#) élément est utilisé pour jouer un effet sonore spécifique à partir d'un fichier son contenant de nombreux effets sonores. Un repère est utilisé pour mettre en pause l'audio, de sorte qu'il se termine exactement à la fin du clip,

même si le navigateur est occupé à exécuter un script. Si la page s'était appuyée sur un script pour mettre en pause l'audio, le début du clip suivant pourrait être entendu si le navigateur n'était pas en mesure d'exécuter le script à l'heure exacte spécifiée.

```
var sfx = new Audio('sfx.wav');
var sounds = sfx.addTextTrack('metadata');

// add sounds we care about
function addFX(start, end, name) {
  var cue = new VTTCue(start, end, '');
  cue.id = name;
  cue.pauseOnExit = true;
  sounds.addCue(cue);
}
addFX(12.783, 13.612, 'dog bark');
addFX(13.612, 15.091, 'kitten mew');

function playSound(id) {
  sfx.currentTime = sounds.getCueById(id).startTime;
  sfx.play();
}

// play a bark as soon as we can
sfx.oncanplaythrough = function () {
  playSound('dog bark');
}

// meow when the user tries to leave,
// and have the browser ask them to stay
window.onbeforeunload = function (e) {
  playSound('kitten mew');
  e.preventDefault();
}
```



[Exposed=Window]

```

interface TextTrackCueList {

    readonly attribute unsigned long length;

    getter TextTrackCue (unsigned long index);

    TextTrackCue? getCueById(DOMString id);

};

```

#### **cuelist.length**

Renvoie le nombre de [cues](#) dans la liste.

#### **cuelist[index]**

Renvoie le [repère de la piste de texte](#) avec index *index* dans la liste. Les repères sont triés dans [l'ordre des repères de piste de texte](#) .

#### **cuelist.getCueById(id)**

Renvoie le premier [repère de piste de texte](#) (dans [l'ordre des repères de piste de texte](#) ) avec [l'identifiant de repère de piste de texte](#) *id* .

Renvoie null si aucune des cues n'a l'identifiant donné ou si l'argument est la chaîne vide.

Un [TextTrackCueList](#) objet représente une liste mise à jour dynamiquement d' [indices de piste de texte](#) dans un ordre donné.



MDN

L' **length** attribut doit retourner le nombre de [cues](#) dans la liste représentée par l' [TextTrackCueList](#) objet.

Les [indices de propriété pris en charge](#) d'un [TextTrackCueList](#) objet à tout instant sont les nombres de zéro au nombre d' [indices](#) dans la liste représentée par l' [TextTrackCueList](#) objet moins un, le cas échéant. S'il n'y a pas [de repères](#) dans la liste, il n'y a pas [d'index de propriété pris en charge](#) .

Pour [déterminer la valeur d'une propriété indexée](#) pour un *index* index donné , l'agent utilisateur doit renvoyer l' *index* ième [piste de texte](#) dans la liste représentée par l' [TextTrackCueList](#) objet.



MDN

La méthode, lorsqu'elle est appelée avec un argument autre que la chaîne vide, doit renvoyer le premier [repère de piste de texte](#) dans la liste représentée par l' objet dont [l'identifiant de repère de piste de texte](#) est *id* , le cas échéant, ou null sinon. Si l'argument est la chaîne vide, la méthode doit renvoyer null. **[getCueById\(id\)](#)** [TextTrackCueList](#)



[Exposed=Window]

```
interface TextTrackCue : EventTarget {
```

```
  readonly attribute TextTrack? track;
```

```
  attribute DOMString id;
```

```
  attribute double startTime;
```

```
  attribute unrestricted double endTime;
```

```
  attribute boolean pauseOnExit;
```

```
  attribute EventHandler onenter;
```

```
  attribute EventHandler onexit;
```

```
};
```

#### **cue.track**

Renvoie l' [TextTrack](#) objet auquel appartient cette [information de piste de texte](#) , le cas échéant, ou null dans le cas contraire.

#### **cue.id** [ = value ]

Renvoie l' [identificateur de repère de la piste de texte](#) .  
Peut être mis en place.

#### **cue.startTime** [ = value ]

Renvoie l' [heure de début du repère de la piste de texte](#) , en secondes.  
Peut être mis en place.

#### **cue.endTime** [ = value ]

Renvoie l' [heure de fin du repère de la piste de texte](#) , en secondes.  
Renvoie l'infini positif pour un [repère de piste de texte illimité](#) .  
Peut être mis en place.

```
cue.pauseOnExit [ = value ]
```

Renvoie vrai si l' [indicateur de pause à la sortie de la piste de texte](#) est défini, faux sinon.

Peut être mis en place.

✓ MDN

L' **track** attribut, lors de l'obtention, doit retourner l' [TextTrack](#) objet de la [piste de texte](#) dans la [liste de repères](#) de laquelle se trouve le [repère de piste de texte](#) que l' objet représente, le cas échéant ; [TextTrackCue](#) ou null dans le cas contraire.

✓ MDN

L' **id** attribut, lors de l'obtention, doit renvoyer l' [identificateur de repère de piste de texte](#) du [repère de piste de texte](#) que l' [TextTrackCue](#) objet représente. Lors du réglage, l' [identificateur de repère de la piste de texte](#) doit être réglé sur la nouvelle valeur.

✓ MDN

L' **startTime** attribut, lors de l'obtention, doit renvoyer l' [heure de début du repère](#) de piste de texte [du repère de piste de texte](#) que l' [TextTrackCue](#) objet représente, en secondes. Lors du réglage, l' [heure de début du repère de la piste de texte](#) doit être réglée sur la nouvelle valeur, interprétée en secondes ; ensuite, si la [piste de texte](#) [TextTrackCue](#) de l'objet se trouve dans la [liste](#) des [pistes de texte](#) d'une piste de texte et que cette [piste de texte](#) se trouve dans la [liste des pistes de texte](#) d'un [élément multimédia](#) et que l' [indicateur d'affichage](#) de l'[élément multimédia](#) n'est pas défini, exécutez le [temps marche à pas](#) pour ça [élément média](#) .

✓ MDN

L' **endTime** attribut, lors de l'obtention, doit renvoyer l' [heure de fin](#) de la piste de texte de la [piste de texte](#) que l' [TextTrackCue](#) objet représente, en secondes ou en infini positif. Lors de la définition, si la nouvelle valeur est Infinity négative ou une valeur Not-a-Number (NaN), lancez une exception [TypeError](#) . Sinon, l' [heure de fin du repère de la piste de texte](#) doit être réglée sur la nouvelle valeur. Ensuite, si la [piste de texte](#) [TextTrackCue](#) de l'objet se trouve dans la [liste](#) des [pistes de texte](#) d'une piste de texte et que cette [piste de texte](#) se trouve dans la [liste des pistes de texte](#) d'un [élément multimédia](#) et que la piste de texte de l' [élément multimédia](#) [afficher l'indicateur d'affiche](#) n'est pas défini, puis exécutez les [marches temporelles sur](#) les étapes pour cet [élément multimédia](#) .

✓ MDN

L' **pauseOnExit** attribut, lors de l'obtention, doit retourner vrai si l' [indicateur de pause à la sortie du repère de piste de texte](#) du [repère de piste de texte](#) que l' [TextTrackCue](#) objet représente est défini ; ou faux sinon. Lors du réglage,

l' [indicateur de pause à la sortie de la piste de texte](#) doit être défini si la nouvelle valeur est vraie et doit être désactivé dans le cas contraire.

#### 4.8.11.11.6 Gestionnaires d'événements pour les objets des API de suivi de texte

Voici les [gestionnaires d'événements](#) qui (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par tous les objets implémentant l' [TextTrackList](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<a href="#">onchange</a>	<a href="#">change</a>
<a href="#">onaddtrack</a>	<a href="#">addtrack</a>
<a href="#">onremovetrack</a>	<a href="#">removetrack</a>

Voici les [gestionnaires d'événements](#) qui (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par tous les objets implémentant l' [TextTrack](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<a href="#">oncuechange</a>	<a href="#">cuechange</a>
✓ MDN	

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par tous les objets implémentant l' [TextTrackCue](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<a href="#">onenter</a>	<a href="#">enter</a>
✓ MDN	
<a href="#">onexit</a>	<a href="#">exit</a>
✓ MDN	

#### 4.8.11.11.7 Meilleures pratiques pour les pistes de texte de métadonnées

Cette section est non normative.



Des pistes de texte peuvent être utilisées pour stocker des données relatives aux données multimédias, pour des vues interactives ou augmentées.

Par exemple, une page affichant une émission sportive peut inclure des informations sur le score actuel. Supposons qu'une compétition de robotique soit diffusée en direct. L'image pourrait être superposée avec les scores, comme suit :

Afin que l'affichage de la partition s'affiche correctement chaque fois que l'utilisateur recherche un point arbitraire dans la vidéo, les repères de piste de texte des métadonnées doivent être aussi longs que nécessaire pour la partition. Par exemple, dans le cadre ci-dessus, il y aurait peut-être un signal qui dure toute la durée du match qui donne le numéro du match, un signal qui dure jusqu'à ce que le score de l'alliance bleue change et un signal qui dure jusqu'à ce que le score de l'alliance rouge change. Si la vidéo n'est qu'un flux de l'événement en direct, l'heure en bas à droite serait probablement automatiquement dérivée de l'heure actuelle de la vidéo, plutôt que basée sur un repère. Cependant, si la vidéo n'était que les faits saillants, cela pourrait également être donné en indices.

Ce qui suit montre à quoi pourraient ressembler des fragments de ceci dans un fichier WebVTT :

WEBVTT

...

05:10:00.000 --> 05:12:15.000  
type de correspondance:qual  
matchnumber:37

...

05:11:02.251 --> 05:11:17.198  
rouge : 78

05:11:03.672 --> 05:11:54.198  
bleu:66

05:11:17.198 --> 05:11:25.912  
rouge:80

05:11:25.912 --> 05:11:26.522  
rouge:83

05:11:26.522 --> 05:11:26.982

rouge:86

05:11:26.982 --> 05:11:27.499

rouge:89

...

La clé ici est de remarquer que l'information est donnée dans des indices qui couvrent la durée à laquelle l'événement pertinent s'applique. Si, à la place, les scores étaient donnés sous forme d'indices de longueur nulle (ou très brefs, presque de longueur nulle) lorsque le score change, par exemple en disant "rouge + 2" à 05:11:17.198, "rouge + 3" à 05:11:25.912, etc., des problèmes surgissent : principalement, la recherche est beaucoup plus difficile à mettre en œuvre, car le script doit parcourir toute la liste des cues pour s'assurer qu'aucune notification n'a été manquée ; mais aussi, si les signaux sont courts, il est possible que le script ne voie jamais qu'ils sont actifs à moins qu'il ne les écoute spécifiquement.

Lors de l'utilisation d'indices de cette manière, les auteurs sont encouragés à utiliser l' [cuechange](#) événement pour mettre à jour les annotations actuelles. (En particulier, utiliser l' [timeupdate](#) événement serait moins approprié car cela nécessiterait de travailler même lorsque les repères n'ont pas changé, et, plus important encore, introduirait une latence plus élevée entre le moment où les repères de métadonnées deviennent actifs et le moment où l'affichage est mis à jour, puisque [timeupdate](#) les événements sont limités en débit.)

#### 4.8.11.12 Identification d'un type de piste via une URL

Les autres spécifications ou formats nécessitant une [URL](#) pour identifier les valeurs de retour des attributs [AudioTrack kind](#) ou [VideoTrack kind](#)IDL , ou identifier le [type de piste de texte](#) , doivent utiliser l' [about:html-kind](#) [URL](#) .

#### 4.8.11.13 Interface utilisateur

L' **controls** attribut est un [attribut booléen](#) . S'il est présent, il indique que l'auteur n'a pas fourni de contrôleur scripté et souhaite que l'agent utilisateur fournisse son propre ensemble de contrôles.

Si l'attribut est présent, ou si [les scripts sont désactivés](#) pour l' [élément média](#) , alors l'agent utilisateur doit **exposer une interface utilisateur à l'utilisateur** . Cette

interface utilisateur doit inclure des fonctionnalités permettant de démarrer la lecture, de mettre la lecture en pause, de rechercher une position arbitraire dans le contenu (si le contenu prend en charge la recherche arbitraire), de modifier le volume, de modifier l'affichage des sous-titres codés ou des pistes en langue des signes intégrées, de sélectionner différentes pistes audio ou activer les descriptions audio, et afficher le contenu multimédia de manière plus adaptée à l'utilisateur (par exemple, vidéo plein écran ou dans une fenêtre redimensionnable indépendante). D'autres contrôles peuvent également être mis à disposition.

Cependant, même lorsque l'attribut est absent, les agents utilisateurs peuvent fournir des commandes pour affecter la lecture de la ressource multimédia (par exemple, lecture, pause, recherche, sélection de piste et commandes de volume), mais ces fonctionnalités ne doivent pas interférer avec le rendu normal de la page. Par exemple, ces fonctionnalités peuvent être exposées dans le menu contextuel de [l'élément multimédia](#), les touches multimédia de la plate-forme ou une télécommande. L'agent utilisateur peut implémenter cela simplement en [exposant une interface utilisateur à l'utilisateur](#) comme décrit ci-dessus (comme si l'`controls` attribut était présent).

Si l'agent utilisateur [expose une interface utilisateur à l'utilisateur](#) en affichant des contrôles sur l'[élément média](#), alors l'agent utilisateur doit supprimer tout événement d'interaction utilisateur pendant que l'agent utilisateur interagit avec cette interface. (Par exemple, si l'utilisateur clique sur le contrôle de lecture d'une vidéo, `mousedown` les événements, etc., ne seront pas déclenchés simultanément sur les éléments de la page.)

Dans la mesure du possible (en particulier, pour démarrer, arrêter, mettre en pause et reprendre la lecture, pour rechercher, pour modifier la vitesse de lecture, pour avancer ou rembobiner rapidement, pour lister, activer et désactiver les pistes de texte, et pour désactiver ou modifier le volume de l'audio), les fonctionnalités de l'interface utilisateur exposées par l'agent utilisateur doivent être implémentées en termes de l'API DOM décrite ci-dessus, de sorte que, par exemple, tous les mêmes événements se déclenchent.

Les fonctionnalités telles que l'avance rapide ou le rembobinage doivent être implémentées en modifiant uniquement l'`playbackRate` attribut (et non l'`defaultPlaybackRate` attribut).

La recherche doit être mise en œuvre en termes de [recherche](#) de la position demandée dans la [chronologie](#) des médias de l'[élément média](#). Pour les ressources multimédias où la recherche d'une position arbitraire serait lente, les agents utilisateurs sont encouragés à utiliser le drapeau *approximatif pour la vitesse* lors de la recherche en réponse à la manipulation par l'utilisateur d'une interface de position approximative telle qu'une barre de recherche.



L'`controls` attribut IDL doit [réfléter](#) l'attribut de contenu du même nom.

---

```
media.volume [ = value ]
```

✓ 

Renvoie le volume de lecture actuel, sous la forme d'un nombre compris entre 0,0 et 1,0, où 0,0 est le plus faible et 1,0 le plus fort.

Peut être réglé, pour changer le volume.

Lance un "[IndexSizeError](#)" [DOMException](#) si la nouvelle valeur n'est pas dans la plage 0.0 .. 1.0.

```
media.muted [ = value ]
```

✓ 

Renvoie true si l'audio est coupé, remplaçant l' [volume](#) attribut, et false si l' [volume](#) attribut est honoré.

Peut être réglé pour changer si le son est coupé ou non.

Un [élément multimédia](#) a un **volume de lecture** , qui est une fraction comprise entre 0,0 (silencieux) et 1,0 (le plus fort). Initialement, le volume devrait être 1.0, mais les agents utilisateurs peuvent se souvenir de la dernière valeur définie à travers les sessions, sur une base par site ou autrement, de sorte que le volume peut commencer à d'autres valeurs.

L' [volume](#) attribut IDL doit renvoyer le [volume de lecture](#) de toutes les parties audio de l' [élément multimédia](#) . Lors du paramétrage, si la nouvelle valeur est comprise entre 0,0 et 1,0 inclus, le [volume de lecture](#) de l' [élément multimédia](#) doit être défini sur la nouvelle valeur. Si la nouvelle valeur est en dehors de la plage de 0,0 à 1,0 inclus, alors, lors de la définition, un " " doit être émis à la place. [IndexSizeError](#) [DOMException](#)

Un [élément multimédia](#) peut également être **mis en sourdine** . Si quelque chose coupe l'élément, alors il est mis en sourdine. (Par exemple, lorsque la [direction de lecture](#) est vers l'arrière, l'élément est mis en sourdine.)

L' [muted](#) attribut IDL doit renvoyer la valeur à laquelle il a été défini en dernier. Lorsqu'un [élément média](#) est créé, si l'élément a un [muted](#) attribut de contenu spécifié, alors l' [muted](#) attribut IDL doit être défini sur vrai ; sinon, les agents utilisateurs peuvent définir la valeur sur la valeur préférée de l'utilisateur (par exemple, se souvenir de la dernière valeur définie à travers les sessions, sur une base par site ou autrement). Lorsque l' [muted](#) attribut IDL est défini sur true, l' [élément média](#) doit être [mis en sourdine](#) .

Chaque fois que l'une ou l'autre des valeurs renvoyées par les attributs IDL [volume](#) et [muted](#) change, l'agent utilisateur doit [mettre en file d'attente une tâche d'élément média](#) en fonction de l' [élément média](#) pour [déclencher un événement](#) nommé [volumechange](#) sur l' [élément média](#) . Ensuite, si l' [élément](#)

[média](#) n'est pas [autorisé à jouer](#) , l'agent utilisateur doit exécuter les [étapes de pause internes](#) pour l' [élément média](#) .

**Le volume média effectif** d'un élément est déterminé comme suit :

1. Si l'utilisateur a indiqué que l'agent utilisateur doit remplacer le volume de l'élément, retournez le volume souhaité par l'utilisateur.
2. Si la sortie audio de l'élément est [muette](#) , renvoie zéro.
3. Soit *volume* le [volume de lecture](#) des parties audio de l' [élément multimédia](#) , compris entre 0,0 (silencieux) et 1,0 (le plus fort).
4. *Volume* de retour , interprété par rapport à la plage de 0,0 à 1,0, 0,0 étant silencieux et 1,0 étant le réglage le plus fort, les valeurs intermédiaires augmentant en volume. La plage n'a pas besoin d'être linéaire. Le réglage le plus fort peut être inférieur au réglage le plus fort possible du système ; par exemple, l'utilisateur aurait pu définir un volume maximum.

L' **muted** attribut content des [éléments multimédias](#) est un [attribut booléen](#) qui contrôle l'état par défaut de la sortie audio de la [ressource multimédia](#) , remplaçant potentiellement les préférences de l'utilisateur.



L' **defaultMuted** attribut IDL doit [refléter](#) l' **muted** attribut content.

*Cet attribut n'a aucun effet dynamique (il ne contrôle que l'état par défaut de l'élément).*

Cette vidéo (une publicité) se lance automatiquement, mais pour éviter d'ennuyer les utilisateurs, elle le fait sans son et permet à l'utilisateur d'activer le son. L'agent utilisateur peut mettre en pause la vidéo si elle est réactivée sans interaction de l'utilisateur.

```
<video src="adverts.cgi?kind=video" controls autoplay loop
muted></video>
```

#### 4.8.11.14 Plages horaires



Les objets implémentant l' [TimeRanges](#) interface représentent une liste de plages (périodes) de temps.

[Exposed=Window]

```
interface TimeRanges {
```

```
    readonly attribute unsigned long length;
```

```
    double start(unsigned long index);
```

```
    double end(unsigned long index);
```

```
};
```

**media.length**

✓

Renvoie le nombre de plages dans l'objet.

**time = media.start(index)**

✓

Renvoie l'heure du début de la plage avec l'index donné.

Lance un "**IndexSizeError**" **DOMException** si l'index est hors limites.

**time = media.end(index)**

✓

Renvoie l'heure de la fin de la plage avec l'index donné.

Lance un "**IndexSizeError**" **DOMException** si l'index est hors limites.

L' **length** attribut IDL doit renvoyer le nombre de plages représentées par l'objet.

La méthode doit renvoyer la position du début de la plage d' *indices* représentée par l'objet, en secondes mesurées à partir du début de la chronologie couverte par l'objet.**start(index)**

La méthode doit renvoyer la position de la fin de la plage d' *indices* représentée par l'objet, en secondes mesurées à partir du début de la chronologie couverte par l'objet.**end(index)**

Ces méthodes doivent lancer des "**IndexSizeError**" **DOMException** si elles sont appelées avec un argument d'*index* supérieur ou égal au nombre de plages représentées par l'objet.

Lorsqu'un **TimeRanges** objet est dit **objet normalisé****TimeRanges** , les plages qu'il représente doivent obéir aux critères suivants :

- Le début d'une plage doit être supérieur à la fin de toutes les plages précédentes.
- Le début d'une plage doit être inférieur ou égal à la fin de cette même plage.

En d'autres termes, les plages d'un tel objet sont ordonnées, ne se chevauchent pas et ne se touchent pas (les plages adjacentes sont pliées en une plage plus grande). Une plage peut être vide (faisant référence à un seul instant dans le temps), par exemple pour indiquer qu'une seule image est actuellement mise en mémoire tampon dans le cas où l'agent utilisateur a supprimé la totalité de la ressource multimédia à l'exception de l'image [actuelle](#), lorsqu'un [élément multimédia](#) est en pause.

Les plages d'un [TimeRanges](#) objet doivent être inclusives.

Ainsi, la fin d'une plage serait égale au début d'une plage adjacente suivante (touchant mais ne se chevauchant pas). De même, une plage couvrant toute une chronologie ancrée à zéro aurait un début égal à zéro et une fin égale à la durée de la chronologie.

Les chronologies utilisées par les objets renvoyés par les attributs [buffered](#), [seekable](#) et [played](#) IDL des [éléments multimédias](#) doivent être [la chronologie multimédia](#) de cet élément.

#### 4.8.11.15 L' [TrackEvent](#) interface



[Exposed=Window]

```
interface TrackEvent : Event {
```

```
  constructor(DOMString type, optional TrackEventInit
```

```
  eventInitDict = {});
```

```
  readonly attribute (VideoTrack or AudioTrack or TextTrack)?
```

```
  track;
```

```
};
```

```
dictionary TrackEventInit : EventInit {
```

```
  (VideoTrack or AudioTrack or TextTrack)? track = null;
```

```
};
```

`event.track`

✓

Renvoie l'objet de piste ( [TextTrack](#), [AudioTrack](#) ou [VideoTrack](#) ) auquel l'événement se rapporte.

L' `track` attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente les informations contextuelles de l'événement.

#### 4.8.11.16 Résumé des événements

*Cette section est non normative.*

Les événements suivants se déclenchent sur [les éléments multimédias](#) dans le cadre du modèle de traitement décrit ci-dessus :

Nom de l'événement	Interface	Licencié quand...	Conditions préalables
<code>loadstart</code> ✓ MDN	<a href="#">Event</a>	L'agent utilisateur commence à rechercher <a href="#">les données multimédias</a> , dans le cadre de l' <a href="#">algorithme de sélection des ressources</a> .	<code>networkState</code> équivaut à <a href="#">NETWORK_LOADING</a>
<code>progress</code> ✓ MDN	<a href="#">Event</a>	L'agent utilisateur récupère <a href="#">les données multimédias</a> .	<code>networkState</code> équivaut à <a href="#">NETWORK_LOADING</a>
<code>suspend</code> ✓ MDN	<a href="#">Event</a>	L'agent utilisateur ne récupère pas actuellement <a href="#">les données multimédias</a> intentionnellement .	<code>networkState</code> équivaut à <a href="#">NETWORK_IDLE</a>
<code>abort</code> ✓ MDN	<a href="#">Event</a>	L'agent utilisateur arrête de récupérer les <a href="#">données multimédias</a> avant qu'elles ne soient complètement téléchargées, mais pas en raison d'une erreur.	<code>error</code> est un objet avec le code <a href="#">MEDIA_ERR_ABORTED</a> . <code>networkState</code> est égal <a href="#">NETWORK_EMPTY</a> à ou <a href="#">NETWORK_IDLE</a> , selon le moment où le téléchargement a été interrompu.
<code>error</code> ✓ MDN	<a href="#">Event</a>	Une erreur se produit lors de la récupération des <a href="#">données multimédias</a> ou le type de ressource n'est pas un format multimédia pris en charge.	<code>error</code> est un objet avec le code <a href="#">MEDIA_ERR_NETWORK</a> ou supérieur. <code>networkState</code> est égal <a href="#">NETWORK_EMPTY</a> à ou <a href="#">NETWORK_IDLE</a> , selon le moment où le téléchargement a été interrompu.



Nom de l'événement	Interface	Licencié quand...	Conditions préalables
<b>emptied</b> ✓ MDN	Event	Un <a href="#">élément média</a> qui <code>networkState</code> n'était pas dans cet <code>NETWORK_EMPTY</code> état vient de basculer dans cet état (soit à cause d'une erreur fatale lors du chargement qui est sur le point d'être signalée, soit parce que la <code>load()</code> méthode a été invoquée alors que l' <a href="#">algorithme de sélection de ressources</a> était déjà en cours d'exécution).	<code>networkState</code> est <code>NETWORK_EMPTY</code> ; tous les attributs IDL sont dans leurs états initiaux.
<b>stalled</b> ✓ MDN	Event	L'agent utilisateur essaie de récupérer <a href="#">les données multimédias</a> , mais les données ne sont pas disponibles de manière inattendue.	<code>networkState</code> est <code>NETWORK_LOADING</code> .
<b>loadedmetadata</b> ✓ MDN	Event	L'agent utilisateur vient de déterminer la durée et les dimensions de la <a href="#">ressource média</a> et <a href="#">les pistes de texte sont prêtes</a> .	<code>readyState</code> est nouvellement égal <code>HAVE_METADATA</code> ou supérieur pour la première fois.
<b>loadeddata</b> ✓ MDN	Event	L'agent utilisateur peut restituer les <a href="#">données multimédias</a> à la <a href="#">position de lecture actuelle</a> pour la première fois.	<code>readyState</code> nouvellement augmenté à <code>HAVE_CURRENT_DATA</code> ou supérieur pour la première fois.
<b>canplay</b> ✓ MDN	Event	L'agent utilisateur peut reprendre la lecture des <a href="#">données multimédias</a> , mais estime que si la lecture devait commencer maintenant, la <a href="#">ressource média</a> ne pourrait pas être restituée à la vitesse de lecture actuelle jusqu'à sa fin sans avoir à s'arrêter pour une mise en mémoire tampon supplémentaire du contenu.	<code>readyState</code> nouvellement augmenté <code>HAVE_FUTURE_DATA</code> ou supérieur.
<b>canplaythrough</b> ✓ MDN	Event	L'agent utilisateur estime que si la lecture devait commencer maintenant, la <a href="#">ressource média</a> pourrait être restituée à la vitesse de lecture actuelle jusqu'à sa fin sans	<code>readyState</code> est nouvellement égal à <code>HAVE_ENOUGH_DATA</code> .

Nom de l'événement	Interface	Licencié quand...	Conditions préalables
		avoir à s'arrêter pour une mise en mémoire tampon supplémentaire.	
<b>playing</b> ✓ MDN	<u>Event</u>	La lecture est prête à démarrer après avoir été interrompue ou retardée en raison d'un manque de <a href="#">données multimédia</a> .	<u>readyState</u> est nouvellement égal ou supérieur à <u>HAVE_FUTURE_DATA</u> et <u>paused</u> est faux, ou <u>paused</u> est nouvellement faux et <u>readyState</u> est égal ou supérieur à <u>HAVE_FUTURE_DATA</u> . Même si cet événement se déclenche, l'élément peut toujours ne pas être <a href="#">en cours de lecture</a> , par exemple si l'élément est <a href="#">mis en pause pour l'interaction de l'utilisateur</a> ou <a href="#">mis en pause pour le contenu intrabande</a> .
<b>waiting</b> ✓ MDN	<u>Event</u>	La lecture s'est arrêtée car la trame suivante n'est pas disponible, mais l'agent utilisateur s'attend à ce que cette trame devienne disponible en temps voulu.	<u>readyState</u> est égal ou inférieur à <u>HAVE_CURRENT_DATA</u> , et <u>paused</u> est faux. Soit <u>seeking</u> est vrai, soit la <a href="#">position de lecture actuelle</a> n'est contenue dans aucune des plages de <u>buffered</u> . Il est possible que la lecture s'arrête pour d'autres raisons sans <u>paused</u> être fausses, mais ces raisons ne déclenchent pas cet événement (et lorsque ces situations se résolvent, un <u>playing</u> événement distinct n'est pas non plus déclenché) : par exemple, <a href="#">la lecture s'est terminée</a> ou la lecture <a href="#">s'est arrêtée en raison d'erreurs</a> , ou l'élément a été <a href="#">mis en pause pour l'interaction de l'utilisateur</a> ou <a href="#">mis en pause pour le contenu intrabande</a> .
<b>seeking</b> ✓ MDN	<u>Event</u>	L' <u>seeking</u> attribut IDL est devenu vrai et l'agent utilisateur	

Nom de l'événement	Interface	Licencié quand...	Conditions préalables
		a commencé à rechercher une nouvelle position.	
<b>seeked</b> ✓ MDN	Event	L' <u>seeking</u> attribut IDL est devenu faux après la modification de la <u>position de lecture actuelle</u> .	
<b>ended</b> ✓ MDN	Event	La lecture s'est arrêtée car la fin de la <u>ressource multimédia</u> a été atteinte.	<u>currentTime</u> est égal à la fin de la <u>ressource média</u> ; <u>ended</u> est vrai.
<b>durationchange</b> ✓ MDN	Event	L' <u>duration</u> attribut vient d'être mis à jour.	
<b>timeupdate</b> ✓ MDN	Event	La <u>position de lecture actuelle</u> a changé dans le cadre de la lecture normale ou d'une manière particulièrement intéressante, par exemple de manière discontinue.	
<b>play</b> ✓ MDN	Event	L'élément n'est plus en pause. Déclenché après <u>play()</u> le retour de la méthode ou lorsque l' <u>autoplay</u> attribut a déclenché la lecture.	<u>paused</u> est nouvellement faux.
<b>pause</b> ✓ MDN	Event	L'élément a été mis en pause. Lancé après le <u>pause()</u> retour de la méthode.	<u>paused</u> est nouvellement vrai.
<b>ratechange</b> ✓ MDN	Event	L' attribut <u>defaultPlaybackRate</u> ou l' <u>playbackRate</u> attribut vient d'être mis à jour.	
<b>resize</b>	Event	L'un ou les deux attributs <u>videoWidth</u> et <u>videoHeight</u> viennent d'être mis à jour.	L' <u>élément média</u> est un <u>video</u> élément ; <u>readyState</u> n'est pas <u>HAVE NOTHING</u>
<b>volumechange</b> ✓ MDN	Event	L' <u>volume</u> attribut ou l' <u>muted</u> attribut a changé. Déclenché après le retour du setter de l'attribut concerné.	

L'événement suivant se déclenche sur source les éléments :

Nom de l'événement	Interface	Licencié quand...
<b>error</b>	<u>Event</u>	Une erreur se produit lors de la récupération des <u>données multimédias</u> ou le type de ressource n'est pas un format multimédia pris en charge.

Les événements suivants se déclenchent sur les objets AudioTrackList, VideoTrackList et TextTrackList:

Nom de l'événement	Interface	Licencié quand...
<b>change</b> ✓ MDN	<u>Event</u>	Une ou plusieurs pistes de la liste des pistes ont été activées ou désactivées.
<b>addtrack</b> ✓ MDN	<u>TrackEvent</u>	Une piste a été ajoutée à la liste des pistes.
<b>removetrack</b> ✓ MDN	<u>TrackEvent</u>	Une piste a été supprimée de la liste des pistes.

L'événement suivant se déclenche sur TextTrack les objets et track les éléments :

Nom de l'événement	Interface	Licencié quand...
<b>cuechange</b> ✓ MDN	<u>Event</u>	Un ou plusieurs repères de la piste sont devenus actifs ou ont cessé de l'être.

Les événements suivants se déclenchent sur track les éléments :

Nom de l'événement	Interface	Licencié quand...
<b>error</b>	<u>Event</u>	Une erreur se produit lors de la récupération des données de piste ou le type de ressource n'est pas pris en charge par le format de piste de texte.
<b>load</b>	<u>Event</u>	Les données d'une piste ont été récupérées et traitées avec succès.

Les événements suivants se déclenchent sur TextTrackCue les objets :

Nom de l'événement	Interface	Licencié quand...
<b>enter</b> ✓ MDN	<u>Event</u>	Le repère est devenu actif.
<b>exit</b> ✓ MDN	<u>Event</u>	La cue a cessé d'être active.

#### 4.8.11.17 Considérations relatives à la sécurité et à la confidentialité

Les principales implications en matière de sécurité et de confidentialité des éléments [video](#) et [audio](#) proviennent de la possibilité d'intégrer des médias d'origine croisée. Les menaces peuvent circuler dans deux directions : du contenu hostile vers une page victime, et d'une page hostile vers le contenu victime.

---

Si une page victime intègre un contenu hostile, la menace est que le contenu peut contenir un code scripté qui tente d'interagir avec celui [Document](#) qui intègre le contenu. Pour éviter cela, les agents utilisateurs doivent s'assurer qu'il n'y a pas d'accès du contenu à la page d'intégration. Dans le cas d'un contenu multimédia qui utilise des concepts DOM, le contenu intégré doit être traité comme s'il se trouvait dans son propre [traversable de niveau supérieur](#) non lié .

Par exemple, si une animation SVG était intégrée dans un [video](#) élément, l'agent utilisateur ne lui donnerait pas accès au DOM de la page externe. Du point de vue des scripts dans la ressource SVG, le fichier SVG semblerait être dans un traversable de niveau supérieur isolé sans parent.

---

Si une page hostile intègre du contenu victime, la menace est que la page d'intégration pourrait obtenir des informations du contenu auxquelles elle n'aurait pas accès autrement. L'API expose certaines informations : l'existence du média, son type, sa durée, sa taille et les caractéristiques de performance de son hôte. Ces informations sont déjà potentiellement problématiques, mais dans la pratique, les mêmes informations peuvent plus ou moins être obtenues à l'aide de l' [img](#) élément, et elles ont donc été jugées acceptables.

Cependant, des informations beaucoup plus sensibles pourraient être obtenues si l'agent utilisateur expose davantage les métadonnées dans le contenu, telles que les sous-titres. Ces informations ne sont donc exposées que si la ressource vidéo utilise CORS. L' [crossorigin](#) attribut permet aux auteurs d'activer CORS. [\[ALLER CHERCHER\]](#)

Sans cette restriction, un attaquant pourrait inciter un utilisateur fonctionnant au sein d'un réseau d'entreprise à visiter un site qui tente de charger une vidéo à partir d'un emplacement précédemment divulgué sur l'intranet de l'entreprise. Si une telle vidéo incluait des plans confidentiels pour un nouveau produit, la possibilité de lire les sous-titres constituerait une grave violation de la confidentialité.

#### 4.8.11.18 Meilleures pratiques pour les auteurs utilisant des éléments multimédias

*Cette section est non normative.*

La lecture de ressources audio et vidéo sur de petits appareils tels que des décodeurs ou des téléphones portables est souvent limitée par des ressources matérielles limitées dans l'appareil. Par exemple, un appareil peut ne prendre en charge que trois vidéos simultanées. Pour cette raison, il est recommandé de libérer les ressources détenues par [les éléments multimédias](#) lorsqu'ils sont terminés, soit en faisant très attention à supprimer toutes les références à l'élément et à lui permettre d'être ramassé, soit, mieux encore, en définissant le paramètre l'attribut de l'élément [src](#) à une chaîne vide. Dans les cas où [srcObject](#) était défini, définissez à la place la [srcObject](#) valeur null.

De même, lorsque le taux de lecture n'est pas exactement de 1,0, des limitations matérielles, logicielles ou de format peuvent entraîner la suppression d'images vidéo et l'audio saccadé ou coupé.

#### 4.8.11.19 Meilleures pratiques pour les implémenteurs d'éléments multimédias

*Cette section est non normative.*

La précision avec laquelle divers aspects de l'API de l' [élément multimédia](#) sont implémentés est considérée comme un problème de qualité d'implémentation.

Par exemple, lors de l'implémentation de l' [buffered](#) attribut, la précision avec laquelle une implémentation signale les plages qui ont été mises en mémoire tampon dépend de la minutie avec laquelle l'agent utilisateur inspecte les données. Étant donné que l'API rapporte des plages d'heures, mais que les données sont obtenues dans des flux d'octets, un agent utilisateur recevant un flux à débit variable peut uniquement être en mesure de déterminer des heures précises en décodant réellement toutes les données. Cependant, les agents utilisateurs ne sont pas tenus de le faire ; ils peuvent à la place renvoyer des estimations (par exemple, basées sur le débit binaire moyen observé jusqu'à présent) qui sont révisées à mesure que davantage d'informations deviennent disponibles.

En règle générale, les agents utilisateurs sont invités à être conservateurs plutôt qu'optimistes. Par exemple, il serait mauvais de signaler que tout a été mis en mémoire tampon alors que ce n'est pas le cas.

Un autre problème de qualité de mise en œuvre serait de lire une vidéo à l'envers lorsque le codec est conçu uniquement pour la lecture vers l'avant (par exemple, il n'y a pas beaucoup d'images clés, et elles sont éloignées, et les images intermédiaires n'ont que des deltas de l'image précédente) . Les agents utilisateurs pourraient faire un mauvais travail, par exemple en n'affichant que les images clés

; cependant, de meilleures implémentations feraient plus de travail et feraient donc un meilleur travail, par exemple en décodant réellement des parties de la vidéo vers l'avant, en stockant les images complètes, puis en lisant les images à l'envers.

De même, bien que les implémentations soient autorisées à supprimer des données mises en mémoire tampon à tout moment (il n'est pas nécessaire qu'un agent utilisateur conserve toutes les données multimédias obtenues pendant la durée de vie de l'élément multimédia), il s'agit à nouveau d'un problème de qualité d'implémentation : les agents utilisateurs avec suffisamment les ressources pour conserver toutes les données sont encouragées à le faire, car cela permet une meilleure expérience utilisateur. Par exemple, si l'utilisateur regarde un flux en direct, un agent utilisateur pourrait autoriser l'utilisateur uniquement à visionner la vidéo en direct ; cependant, un meilleur agent utilisateur mettrait tout en mémoire tampon et permettrait à l'utilisateur de parcourir le matériel précédent, de le mettre en pause, de le lire en avant et en arrière, etc.

---

Lorsqu'un [élément multimédia](#) mis en pause est [supprimé d'un document](#) et n'est pas réinséré avant la prochaine fois que la [boucle d'événements](#) atteint [l'étape 1](#) , les implémentations limitées en ressources sont encouragées à saisir cette opportunité pour libérer toutes les ressources matérielles (comme les plans vidéo, les ressources réseau, et tampons de données) utilisés par l' [élément média](#) . (Les agents utilisateurs doivent toujours garder une trace de la position de lecture et ainsi de suite, au cas où la lecture serait redémarrée ultérieurement.)

#### 4.8.12 L' **map**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Transparente](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

### Attributs globaux

name — Nom de l'image cliquable à référer à partir de l' usemap attribut

## Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

## Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLMapElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute DOMString name;
```

```
  [SameObject] readonly attribute HTMLCollection areas;
```

```
};
```

L' map élément, en conjonction avec un img élément et tout area descendant d'élément, définit une image cliquable . L'élément représente ses enfants.

L' name attribut donne un nom à la carte afin qu'elle puisse être référéncée . L'attribut doit être présent et doit avoir une valeur non vide sans espace ASCII . La valeur de l' name attribut ne doit pas être égale à la valeur de l' name attribut d'un autre map élément du même arbre . Si l' id attribut est également spécifié, les deux attributs doivent avoir la même valeur.

## map.areas

Renvoie un HTMLCollection des area éléments du map.

L' areas attribut doit renvoyer une HTMLCollection racine à l' map élément, dont le filtre correspond uniquement area aux éléments.

L'attribut IDL name doit refléter l'attribut de contenu du même nom.

Les images cliquables peuvent être définies conjointement avec d'autres contenus de la page, pour faciliter la maintenance. Cet exemple est celui d'une page avec une image cliquable en haut de la page et un ensemble correspondant de liens texte en bas.

```
<!DOCTYPE HTML>
<HTML LANG="EN">
```



```

<TITLE>Babies™: Toys</TITLE>
<HEADER>
  <H1>Toys</H1>
  <IMG SRC="/images/menu.gif"
    ALT="Babies™ navigation menu. Select a department to go to
    its page."
    USEMAP="#NAV">
</HEADER>
...
<FOOTER>
  <MAP NAME="NAV">
    <P>
      <A HREF="/clothes/">Clothes</A>
      <AREA ALT="Clothes" COORDS="0,0,100,50" HREF="/clothes/"> |
      <A HREF="/toys/">Toys</A>
      <AREA ALT="Toys" COORDS="100,0,200,50" HREF="/toys/"> |
      <A HREF="/food/">Food</A>
      <AREA ALT="Food" COORDS="200,0,300,50" HREF="/food/"> |
      <A HREF="/books/">Books</A>
      <AREA ALT="Books" COORDS="300,0,400,50" HREF="/books/">
    </P>
  </MAP>
</FOOTER>

```

#### 4.8.13 L' **area** élément



##### Catégories :

Contenu du flux .

Contenu de la phrase .

##### Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de phrase est attendu, mais seulement s'il existe un map ancêtre d'élément.

##### Modèle de contenu :

Rien .

##### Omission de balise dans text/html :

Pas de balise de fin .

## Attributs de contenu :

### Attributs globaux

alt— Texte de remplacement à utiliser lorsque les images ne sont pas disponibles

coords— Coordonnées de la forme à créer dans une [image cliquable](#)

shape— Le type de forme à créer dans une [image cliquable](#)

href— Adresse du [lien hypertexte](#)

target— [Navigable](#) pour [la navigation par lien hypertexte](#)

download— S'il faut télécharger la ressource au lieu d'y accéder, et son nom de fichier si c'est le cas

ping— [URL](#) à pinger

rel— Relation entre l'emplacement dans le document contenant l' [hyperlien](#) et la ressource de destination

referrerpolicy— [Politique de référence](#) pour [les récupérations](#) initiées par l'élément

## Considérations d'accessibilité :

Si l'élément a un hrefattribut : [pour les auteurs](#) ; [pour les exécutants](#) .

Sinon : [pour les auteurs](#) ; [pour les exécutants](#) .

## Interface DOM :

[Exposed=Window]

interface **HTMLAreaElement** : [HTMLElement](#) {

[[HTMLConstructor](#)] constructor();

[[CEReactions](#)] attribute DOMString [alt](#);

[[CEReactions](#)] attribute DOMString [coords](#);

[[CEReactions](#)] attribute DOMString [shape](#);

[[CEReactions](#)] attribute DOMString [target](#);

[[CEReactions](#)] attribute DOMString [download](#);

[[CEReactions](#)] attribute USVString [ping](#);

[[CEReactions](#)] attribute DOMString [rel](#);

[SameObject, PutForwards=[value](#)] readonly attribute

[DOMTokenList](#) [relList](#);

```
[CEReactions] attribute DOMString referrerPolicy;  
  
// also has obsolete members  
  
};  
  
HTMLAreaElement includes HTMLHyperlinkElementUtils;
```

L' [area](#)élément [représente](#) soit un lien hypertexte avec du texte et une zone correspondante sur une [image cliquable](#) , soit une zone morte sur une image cliquable.

Un [area](#)élément avec un nœud parent doit avoir un [map](#)ancêtre d'élément.

Si l' [area](#)élément a un [href](#) attribut, alors l' [area](#)élément représente un [lien hypertexte](#) . Dans ce cas, l' [alt](#)attribut doit être présent. Il spécifie le texte du lien hypertexte. Sa valeur doit être un texte qui, lorsqu'il est présenté avec les textes spécifiés pour les autres hyperliens de l' [image cliquable](#) et avec le texte alternatif de l'image, mais sans l'image elle-même, offre à l'utilisateur le même type de choix que l'hyperlien le ferait lorsqu'il est utilisé sans son texte mais avec sa forme appliquée à l'image. L' [alt](#)attribut peut être laissé vide s'il existe un autre [area](#)élément dans la même [image cliquable](#)[alt](#) qui pointe vers la même ressource et a un attribut non vide .

Si l' [area](#)élément n'a pas [href](#) d'attribut, la zone représentée par l'élément ne peut pas être sélectionnée et l' [alt](#)attribut doit être omis.

Dans les deux cas, les attributs [shape](#)et [coords](#)spécifient la zone.

L' [shape](#)attribut est un [attribut énuméré](#) . Le tableau suivant répertorie les mots clés définis pour cet attribut. Les états donnés dans la première cellule des lignes avec des mots-clés donnent les états auxquels ces mots-clés correspondent. Certains des mots clés ne sont pas conformes, comme indiqué dans la dernière colonne.

État	Mots clés	Remarques
<a href="#">État du cercle</a>	<a href="#">circle</a>	
	<a href="#">circ</a>	Non conforme
<a href="#">État par défaut</a>	<a href="#">default</a>	
<a href="#">État du polygone</a>	<a href="#">poly</a>	
	<a href="#">polygon</a>	Non conforme
<a href="#">État rectangulaire</a>	<a href="#">rect</a>	
	<a href="#">rectangle</a>	Non conforme

L'attribut peut être omis. La [valeur par défaut manquante](#) et [la valeur par défaut invalide](#) sont l'état du [rectangle](#).

L' [coords](#) attribut doit, s'il est spécifié, contenir une [liste valide de nombres à virgule flottante](#). Cet attribut donne les coordonnées de la forme décrite par l' [shape](#) attribut. Le traitement de cet attribut est décrit dans le cadre du modèle de traitement [de carte d'image](#).

Dans l' **état du cercle**, [area](#)les éléments doivent avoir un [coords](#)attribut présent, avec trois entiers, dont le dernier doit être non négatif. Le premier entier doit être la distance en [pixels CSS](#) entre le bord gauche de l'image et le centre du cercle, le deuxième entier doit être la distance en [pixels CSS](#) entre le bord supérieur de l'image et le centre du cercle, et le troisième entier doit être le rayon du cercle, toujours en [pixels CSS](#).

Dans l' **état par défaut**, [area](#)les éléments ne doivent pas avoir d' [coords](#) attribut. (La zone est l'image entière.)

Dans l' **état polygone**, [area](#)les éléments doivent avoir un [coords](#)attribut avec au moins six entiers, et le nombre d'entiers doit être pair. Chaque paire d'entiers doit représenter une coordonnée donnée comme les distances de la gauche et du haut de l'image en [pixels CSS](#) respectivement, et toutes les coordonnées ensemble doivent représenter les points du polygone, dans l'ordre.

Dans l' **état rectangle**, [area](#)les éléments doivent avoir un [coords](#)attribut avec exactement quatre entiers, dont le premier doit être inférieur au troisième et le second doit être inférieur au quatrième. Les quatre points doivent représenter respectivement la distance entre le bord gauche de l'image et le côté gauche du rectangle, la distance entre le bord supérieur et le côté supérieur, la distance entre le bord gauche et le côté droit, et la distance du bord supérieur au côté inférieur, le tout en [pixels CSS](#).

Lorsque les agents utilisateurs permettent aux utilisateurs de [suivre des hyperliens](#) ou [de télécharger des hyperliens](#) créés à l'aide de l' [area](#)élément, comme décrit dans la section suivante, les [href](#)attributs [target](#), [download](#), et [ping](#) décident de la façon dont le lien est suivi. L' [rel](#) attribut peut être utilisé pour indiquer à l'utilisateur la nature probable de la ressource cible avant que l'utilisateur ne suive le lien.

Les attributs [target](#), [download](#), [ping](#), [rel](#) et [referrerpolicy](#) doivent être omis si l' [href](#)attribut n'est pas présent.

Si l' [itemprop](#)attribut est spécifié sur un [area](#)élément, alors l' [href](#)attribut doit également être spécifié.

Le [comportement d'activation](#) d'un [élément area](#) element est :

1. Si l'élément n'a pas [href](#)d'attribut, alors retournez.

2. Si *element* a un [download](#) attribut, ou si l'utilisateur a exprimé une préférence pour télécharger le lien hypertexte, alors [téléchargez le lien hypertexte](#) créé par *element*.
3. Sinon, [suivez le lien hypertexte](#) créé par *element*.



Les attributs IDL [alt](#), [coords](#), [target](#), [download](#), [ping](#) et [rel](#), doivent chacun [refléter](#) les attributs de contenu respectifs du même nom.

L'attribut IDL [shape](#) doit [refléter](#) l' [shape](#) attribut content.



L'attribut IDL [relList](#) doit [refléter](#) l' [rel](#) attribut content.



L'attribut IDL [referrerPolicy](#) doit [refléter](#) l' [referrerpolicy](#) attribut content, [limité aux seules valeurs connues](#).

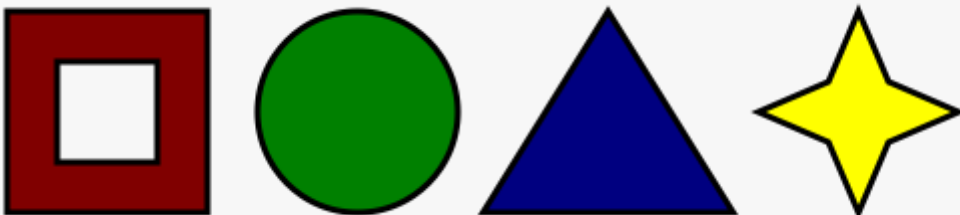
## 4.8.14 Images cliquables

### 4.8.14.1 Création

Une **image cliquable** permet d'associer des zones géométriques d'une image à [des hyperliens](#).

Une image, sous forme d' [img](#) élément, peut être associée à une image cliquable (sous forme d' [map](#) élément) en spécifiant un [usemap](#) attribut sur l' [img](#) élément. L' [usemap](#) attribut, s'il est spécifié, doit être une [référence de nom de hachage valide](#) à un [map](#) élément.

Considérez une image qui ressemble à ceci :



Si nous voulions que seules les zones colorées soient cliquables, nous pourrions le faire comme suit :

```
<p>
  Please select a shape:
  
  <map name="shapes">
    <area shape=rect coords="50,50,100,100"> <!-- the hole in the red
    box -->
    <area shape=rect coords="25,25,125,125" href="red.html" alt="Red
    box.">
    <area shape=circle coords="200,75,50" href="green.html"
    alt="Green circle.">
    <area shape=poly coords="325,25,262,125,388,125" href="blue.html"
    alt="Blue triangle.">
    <area shape=poly
    coords="450,25,435,60,400,75,435,90,450,125,465,90,500,75,465,60"
    href="yellow.html" alt="Yellow star.">
  </map>
</p>
```

#### 4.8.14.2 Modèle de traitement

Si un imgélément a un usemap attribut spécifié, les agents utilisateurs doivent le traiter comme suit :

1. Analysez la valeur de l'attribut en utilisant les [règles d'analyse d'une référence de nom de hachage](#) à un mapélément, avec l'élément comme nœud de contexte. Cela renverra soit un élément (la *map*) soit null.
2. Si cela retourne null, alors retournez. L'image n'est pas associée à une image cliquable après tout.
3. Sinon, l'agent utilisateur doit collecter tous les areaéléments descendants de la *map*. Que ce soient les *domaines*.

Après avoir obtenu la liste des areaéléments qui forment l'image cliquable (les *zones*), les agents utilisateurs interactifs doivent traiter la liste de deux manières.

Si l'agent utilisateur a l'intention d'afficher le texte imgreprésenté par l'élément, il doit suivre les étapes suivantes.

1. Supprimez tous les [area](#)éléments dans les zones qui n'ont pas [href](#)d'attribut.
2. Supprimez tous les [area](#)éléments dans les zones qui n'ont pas [alt](#)d'attribut, ou dont [alt](#) la valeur de l'attribut est la chaîne vide, s'il existe un autre [area](#)élément dans les zones avec la même valeur dans l'attribut et avec un attribut [href](#)non vide [alt](#)
3. [area](#)Chaque élément restant dans les zones représente un [lien hypertexte](#) . Ces hyperliens doivent tous être mis à la disposition de l'utilisateur d'une manière associée au texte du [img](#).

Dans ce contexte, les agents utilisateurs peuvent représenter [area](#)et [img](#)des éléments sans [alt](#)attributs spécifiés, ou dont [alt](#) les attributs sont la chaîne vide ou un autre texte non visible, d'une manière [définie par l'implémentation](#) destinée à indiquer le manque de texte approprié fourni par l'auteur.

Si l'agent utilisateur a l'intention d'afficher l'image et de permettre l'interaction avec l'image pour sélectionner des hyperliens, alors l'image doit être associée à un ensemble de formes en couches, tirées des éléments dans les zones , [area](#)dans l'ordre inverse [de l'arborescence](#) (donc le dernier [area](#)élément spécifié dans la *carte* est la forme la plus basse et le premier élément de la *carte* , dans [l'ordre de l'arborescence](#) , est la forme la plus haute).

Chaque [area](#)élément des zones doit être traité comme suit pour obtenir une forme à superposer sur l'image :

1. Trouvez l'état [shape](#)représenté par l'attribut de l'élément.
2. Utilisez les [règles d'analyse d'une liste de nombres à virgule flottante](#) pour analyser l' [coords](#)attribut de l'élément, s'il est présent, et laissez le résultat être la liste *de coordonnées* . Si l'attribut est absent, laissez la liste *des coordonnées* être la liste vide.
3. Si le nombre d'éléments dans la liste *des coordonnées* est inférieur au nombre minimum donné pour l' [area](#)état actuel de l'élément, conformément au tableau suivant, la forme est vide ; retour.

État	Nombre minimal d'articles
<a href="#">État du cercle</a>	3
<a href="#">État par défaut</a>	0
<a href="#">État du polygone</a>	6
<a href="#">État rectangulaire</a>	4

4. Vérifiez les éléments en excès dans la liste *des coordonnées* conformément à l'entrée dans la liste suivante correspondant à l' [shape](#)état de l'attribut :

### État du cercle

Déposez tous les éléments de la liste au-delà du troisième.

### État par défaut

Déposez tous les éléments de la liste.

### État du polygone

Déposez le dernier élément s'il y a un nombre impair d'éléments.

### État rectangulaire

Déposez tous les éléments de la liste au-delà du quatrième.

5. Si l' shape attribut représente l' état du rectangle et que le premier nombre de la liste est numériquement supérieur au troisième nombre de la liste, échangez ces deux nombres.
6. Si l' shape attribut représente l' état du rectangle et que le deuxième nombre de la liste est numériquement supérieur au quatrième nombre de la liste, échangez ces deux nombres.
7. Si l' shape attribut représente l' état du cercle et que le troisième nombre de la liste est inférieur ou égal à zéro, la forme est vide ; retour.
8. Or, la forme représentée par l'élément est celle décrite pour l'entrée dans la liste ci-dessous correspondant à l'état de l' shape attribut :

### État du cercle

Soit  $x$  le premier nombre de *coords* ,  $y$  le deuxième nombre et  $r$  le troisième nombre.

La forme est un cercle dont le centre est à  $x$  pixels CSS du bord gauche de l'image et à  $y$  pixels CSS du bord supérieur de l'image, et dont le rayon est de  $r$  pixels CSS .

### État par défaut

La forme est un rectangle qui couvre exactement toute l'image.

### État du polygone

Soit  $x_i$  la  $(2i)$  ième entrée de *coords* , et  $y_i$  la  $(2i+1)$  ième entrée de *coords* (la première entrée de *coords* étant celle d'indice 0).

Soit les coordonnées  $(x_i, y_i)$ , interprétées en pixels CSS mesurés à partir du haut à gauche de l'image, pour toutes les valeurs entières de  $i$  de 0 à  $(N/2)-1$  , où  $N$  est le nombre d'éléments dans *coordonnées* .

La forme est un polygone dont les sommets sont donnés par les coordonnées , et dont l'intérieur est établi selon la règle pair-impair. [\[GRAPHIQUE\]](#)

### État rectangulaire



Soit  $x_1$  le premier nombre dans *coords*,  $y_1$  le deuxième nombre,  $x_2$  le troisième nombre et  $y_2$  le quatrième nombre.

La forme est un rectangle dont le coin supérieur gauche est donné par la coordonnée  $(x_1, y_1)$  et dont le coin inférieur droit est donné par la coordonnée  $(x_2, y_2)$ , ces coordonnées étant interprétées comme [des pixels CSS](#) à partir du haut coin gauche de l'image.

Pour des raisons historiques, les coordonnées doivent être interprétées par rapport à l'image *affichée* après tout étirement causé par les propriétés CSS `'width'` et `'height'` (ou, pour les navigateurs non-CSS, les éléments `width` et `height` les attributs de l'image — les navigateurs CSS mappent ces attributs à les propriétés CSS susmentionnées).

*Les fonctionnalités de zoom du navigateur et les transformations appliquées à l'aide de CSS ou de SVG n'affectent pas les coordonnées.*

L'interaction du dispositif de pointage avec une image associée à un ensemble de formes en couches selon l'algorithme ci-dessus doit entraîner le déclenchement des événements d'interaction utilisateur pertinents vers la forme la plus élevée couvrant le point que le dispositif de pointage a indiqué, le cas échéant, ou vers l'image élément lui-même, s'il n'y a pas de forme couvrant ce point. Les agents utilisateurs peuvent également permettre à des éléments individuels `area` représentant [des hyperliens](#) d'être sélectionnés et activés (par exemple à l'aide d'un clavier).

*Comme un `map` élément (et ses `area` éléments) peuvent être associés à plusieurs `img` éléments, il est possible qu'un `area` élément corresponde à plusieurs [zones focalisables](#) du document.*

Les images cliquables sont [en direct](#) ; si le DOM est muté, alors l'agent utilisateur doit agir comme s'il avait réexécuté les algorithmes pour les images cliquables.

#### 4.8.15 MathML



L'élément [MathML`math`](#) appartient aux catégories [de contenu intégré](#), [de contenu de formulation](#), [de contenu de flux](#) et [de contenu palpable](#) pour les besoins des modèles de contenu de cette spécification.

Lorsque l'élément [MathML`annotation-xml`](#) contient des éléments de l'[espace de noms HTML](#), ces éléments doivent tous être [du contenu de flux](#).

Lorsque les éléments de jeton MathML (`mi`, `mo`, `mn`, `ms` et `mtext`) sont des descendants d'éléments HTML, ils peuvent contenir des éléments [de contenu de phrasé](#) provenant de l'[espace de noms HTML](#).

Les agents utilisateurs doivent gérer le texte autre que [les espaces blancs inter-éléments](#) trouvés dans les éléments MathML dont les modèles de contenu n'autorisent pas le texte droit en prétendant, pour les besoins des modèles de contenu MathML, de la mise en page et du rendu, que le texte est en fait enveloppé dans un élément MathML [.mtext](#) (Ce texte n'est cependant pas conforme.)

Les agents utilisateurs doivent agir comme si tout élément MathML dont le contenu ne correspond pas au modèle de contenu de l'élément était remplacé, pour les besoins de la mise en page et du rendu MathML, par un élément [MathML merror](#) contenant un message d'erreur approprié.

La sémantique des éléments MathML est définie par *MathML* et [d'autres spécifications applicables](#) . [\[MATHML\]](#)

Voici un exemple d'utilisation de MathML dans un document HTML :

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>The quadratic formula</title>
  </head>
  <body>
    <h1>The quadratic formula</h1>
    <p>
      <math>
        <mi>x</mi>
        <mo>=</mo>
        <mfrac>
          <mrow>
            <mo form="prefix">-</mo> <mi>b</mi>
            <mo>±</mo>
            <msqrt>
              <msup> <mi>b</mi> <mn>2</mn> </msup>
              <mo>-</mo>
              <mn>4</mn> <mo>×</mo> <mi>a</mi> <mo>×</mo> <mi>c</mi>
            </msqrt>
          </mrow>
          <mrow>
            <mn>2</mn> <mo>×</mo> <mi>a</mi>
          </mrow>
        </mfrac>
      </math>
    </p>
```

```
</body>
</html>
```

#### 4.8.16 SVG



L'élément [SVG<sub>svg</sub>](#) tombe dans les catégories [de contenu intégré](#) , [de contenu de formulation](#) , [de contenu de flux](#) et [de contenu palpable](#) pour les besoins des modèles de contenu de cette spécification.

Lorsque l'élément [SVG<sub>foreignObject</sub>](#) contient des éléments de l' [espace de noms HTML](#) , ces éléments doivent tous être [du contenu de flux](#) .

Le modèle de contenu pour l'élément [SVG à l'intérieur <sub>title</sub>des documents HTML](#) est [le phrasing content](#) . (Ceci limite davantage les exigences données dans SVG 2 .)

La sémantique des éléments SVG est définie par SVG 2 et [d'autres spécifications applicables](#) . [\[SVG\]](#)

---

```
doc = iframe.getSVGDocument()
doc = embed.getSVGDocument()
doc = object.getSVGDocument()
```

Renvoie l' [Document](#) objet, dans le cas des éléments [iframe](#), [embed](#), ou [object](#) utilisés pour incorporer SVG.

Les [getSVGDocument\(\)](#) étapes de la méthode sont :

1. Soit *document* le contenu de [ce document](#) .
2. Si *document* n'est pas nul et a été créé par le [modèle de traitement de chargement de page pour](#) la section des fichiers XML parce que le [type calculé de la ressource](#) dans l' algorithme [de navigation <sub>image/svg+xml</sub>](#) était , alors renvoyez *document* .
3. Renvoie nul.

#### 4.8.17 Attributs de dimension

**Prérequis de l'auteur :** Les

attributs `width` et `height` sur `img`, `iframe`, `embed`, `object`, `video`, `source` lorsque le parent est un `picture` élément et, lorsque leur `type` attribut est à l'état `Image Button`, `input` les éléments peuvent être spécifiés pour donner les dimensions du contenu visuel de l'élément (la largeur et la hauteur respectivement, par rapport à la direction nominale du support de sortie), en `pixels CSS`. Les attributs, s'ils sont spécifiés, doivent avoir des valeurs qui sont des entiers non négatifs valides.

Les dimensions spécifiées données peuvent différer des dimensions spécifiées dans la ressource elle-même, car la ressource peut avoir une résolution différente de la résolution en pixels CSS. (Sur les écrans, les pixels CSS ont une résolution de 96 ppi, mais en général, la résolution des pixels CSS dépend de la distance de lecture.) Si les deux attributs sont spécifiés, l'une des affirmations suivantes doit être vraie :

- $\text{largeur spécifiée} - 0,5 \leq \text{hauteur spécifiée} * \text{rapport cible} \leq \text{largeur spécifiée} + 0,5$
- $\text{hauteur spécifiée} - 0,5 \leq \text{largeur spécifiée} / \text{rapport cible} \leq \text{hauteur spécifiée} + 0,5$
- $\text{hauteur spécifiée} = \text{largeur spécifiée} = 0$

Le *ratio cible* est le rapport de la largeur intrinsèque à la hauteur intrinsèque dans la ressource. La *largeur* et la *hauteur spécifiées* sont respectivement les valeurs des attributs `width` et `height`.

Les deux attributs doivent être omis si la ressource en question n'a pas à la fois une largeur intrinsèque et une hauteur intrinsèque.

Si les deux attributs sont tous les deux nuls, cela indique que l'élément n'est pas destiné à l'utilisateur (par exemple, il peut faire partie d'un service pour compter les pages vues).

*Les attributs de dimension ne sont pas destinés à être utilisés pour étirer l'image.*

**Exigences de l'agent utilisateur :** Les agents utilisateurs sont censés utiliser ces attributs comme conseils pour le rendu.



Les attributs `width` et `height` IDL des éléments `iframe`, `embed`, `object`, `source` et `video` doivent réfléter les attributs de contenu respectifs du même nom.

*Pour `iframe`, `embed` et `object` les attributs IDL sont `DOMString`; pour `video` et `source` les attributs IDL sont `unsigned long`.*

Les attributs IDL correspondants pour les éléments [img](#) et [input](#) sont définis dans les sections de ces éléments respectifs, car ils sont légèrement plus spécifiques aux autres comportements de ces éléments.

## 4.9 Données tabulaires

### 4.9.1 L' `table` élément



#### Catégories :

[Contenu du flux](#) .  
[Contenu palpable](#) .

#### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de flux](#) est attendu.

#### Modèle de contenu :

Dans cet ordre : éventuellement un [caption](#) élément, suivi de zéro ou plusieurs [colgroup](#) éléments, suivi éventuellement d'un [thead](#) élément, suivi de zéro ou plusieurs [tbody](#) éléments ou d'un ou plusieurs [tr](#) éléments, suivi éventuellement d'un [tfoot](#) élément, éventuellement mélangé à un ou plusieurs [éléments de support de script](#) .

#### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

#### Attributs de contenu :

[Attributs globaux](#)

#### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

#### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLTableElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute HTMLTableCaptionElement? caption;
```

```

HTMLTableCaptionElement createCaption();

[CEReactions] undefined deleteCaption();


[CEReactions] attribute HTMLTableSectionElement? tHead;

HTMLTableSectionElement createTHead();

[CEReactions] undefined deleteTHead();


[CEReactions] attribute HTMLTableSectionElement? tFoot;

HTMLTableSectionElement createTFoot();

[CEReactions] undefined deleteTFoot();


[SameObject] readonly attribute HTMLCollection tBodies;

HTMLTableSectionElement createTBody();


[SameObject] readonly attribute HTMLCollection rows;

HTMLTableRowElement insertRow(optional long index = -1);

[CEReactions] undefined deleteRow(long index);


// also has obsolete members

};

```

L' table élément représente des données à plusieurs dimensions, sous la forme d'un tableau .

L' table élément participe au modèle de table . Les tableaux ont des lignes, des colonnes et des cellules données par leurs descendants. Les lignes et les colonnes forment une grille ; les cellules d'un tableau doivent couvrir complètement cette grille sans se chevaucher.

*Des règles précises pour déterminer si cette exigence de conformité est satisfaite sont décrites dans la description du [modèle de table](#) .*

Les auteurs sont encouragés à fournir des informations décrivant comment interpréter les tableaux complexes. Des conseils sur la manière de [fournir ces informations](#) sont donnés ci-dessous.

Les tableaux ne doivent pas être utilisés comme aides à la mise en page. Historiquement, certains auteurs Web ont abusé des tableaux en HTML pour contrôler la mise en page de leur page. Cet usage est non conforme, car les outils tentant d'extraire des données tabulaires de tels documents obtiendraient des résultats très confus. En particulier, les utilisateurs d'outils d'accessibilité tels que les lecteurs d'écran auront probablement beaucoup de mal à naviguer dans les pages avec des tableaux utilisés pour la mise en page.

*Il existe une variété d'alternatives à l'utilisation de tableaux HTML pour la mise en page, telles que la mise en page de grille CSS, la mise en page de boîte flexible CSS ("flexbox"), la mise en page multi-colonnes CSS, le positionnement CSS et le modèle de table CSS. [\[CSS\]](#)*

---

Les tableaux peuvent être compliqués à comprendre et à naviguer. Pour aider les utilisateurs dans cette tâche, les agents utilisateurs doivent clairement délimiter les cellules d'un tableau les unes des autres, à moins que l'agent utilisateur n'ait classé le tableau comme tableau de mise en page (non conforme).

*Les auteurs et les implémenteurs sont encouragés à envisager d'utiliser certaines des [techniques de conception de tableaux](#) décrites ci-dessous pour faciliter la navigation dans les tableaux pour les utilisateurs.*

Les agents utilisateurs, en particulier ceux qui effectuent une analyse de table sur un contenu arbitraire, sont encouragés à trouver des heuristiques pour déterminer quelles tables contiennent réellement des données et lesquelles sont simplement utilisées pour la mise en page. Cette spécification ne définit pas une heuristique précise, mais les éléments suivants sont suggérés comme indicateurs possibles :

Fonctionnalité	Indication
L'utilisation de l' <a href="#">role</a> attribut avec la valeur <a href="#">presentation</a>	Probablement un tableau de disposition
L'utilisation de l'attribut non conforme <a href="#">border</a> avec la valeur non conforme 0	Probablement un tableau de disposition
L'utilisation des attributs non conformes <a href="#">cellspacing</a> et <a href="#">cellpadding</a> avec la valeur 0	Probablement un tableau de disposition

Fonctionnalité	Indication
L'utilisation des éléments <u>caption</u> , <u>thead</u> , ou <u>th</u>	Probablement une table sans mise en page
L'utilisation des attributs <u>headerset</u> <u>scope</u>	Probablement une table sans mise en page
L'utilisation de l'attribut non conforme <u>border</u> avec une valeur autre que 0	Probablement une table sans mise en page
Bordures visibles explicites définies à l'aide de CSS	Probablement une table sans mise en page
L'utilisation de l' <u>summary</u> attribut	Ce n'est pas un bon indicateur (les tableaux avec et sans mise en page ont historiquement reçu cet attribut)

*Il est tout à fait possible que les suggestions ci-dessus soient fausses. Les implémenteurs sont invités à fournir des commentaires sur leurs expériences en essayant de créer une heuristique de détection de table de mise en page.*

Si un table élément a un attribut (non conforme) summary et que l'agent utilisateur n'a pas classé le tableau comme tableau de mise en page, l'agent utilisateur peut signaler le contenu de cet attribut à l'utilisateur.

```
table.caption [ = value ]
```

✓ 

Renvoie l'élément du tableau caption.

Peut être réglé, pour remplacer l' caption élément.

```
caption = table.createCaption()
```

✓ 

Vérifie que la table a un caption élément et le renvoie.

```
table.deleteCaption()
```

✓ 

Garantit que la table n'a pas d' caption élément.

```
table.tHead [ = value ]
```

✓ 

Renvoie l'élément du tableau thead.



Peut être réglé, pour remplacer l' theadélément. Si la nouvelle valeur n'est pas un theadélément, lance un "HierarchyRequestError" DOMException .

```
thead = table.createThead()
```

✓

Vérifie que la table a un theadélément et le renvoie.

```
table.deleteThead()
```

✓

Garantit que la table n'a pas d' theadélément.

```
table.tFoot [ = value ]
```

✓

Renvoie l'élément du tableau tfoot.

Peut être réglé, pour remplacer l' tfootélément. Si la nouvelle valeur n'est pas un tfootélément, lance un "HierarchyRequestError" DOMException .

```
tfoot = table.createTFoot()
```

✓

Vérifie que la table a un tfootélément et le renvoie.

```
table.deleteTFoot()
```

✓

Garantit que la table n'a pas d' tfootélément.

```
table.tBodies
```

✓

Renvoie un HTMLCollectiondes tbodyéléments du tableau.

```
tbody = table.createTBody()
```

✓

Crée un tbodyélément, l'insère dans le tableau et le renvoie.

```
table.rows
```

✓

Renvoie un HTMLCollectiondes tréléments du tableau.

```
tr = table.insertRow([ index ])
```

✓

Crée un trélément, avec un tbodysi nécessaire, les insère dans le tableau à la position donnée par l'argument et renvoie le tr.

La position est relative aux lignes du tableau. L'indice -1, qui est la valeur par défaut si l'argument est omis, équivaut à insérer à la fin du tableau.

Si la position donnée est inférieure à -1 ou supérieure au nombre de lignes, lance un `"IndexSizeError" DOMException`.

`table.deleteRow(index)`

✓

Supprime l'trélément avec la position donnée dans le tableau.

La position est relative aux lignes du tableau. L'indice -1 équivaut à supprimer la dernière ligne du tableau.

Si la position donnée est inférieure à -1 ou supérieure à l'index de la dernière ligne, ou s'il n'y a pas de ligne, lance un `"IndexSizeError" DOMException`.

Dans toutes les définitions d'attributs et de méthodes suivantes, lorsqu'un élément doit être **créé dans une table**, cela signifie créer un élément en fonction du nœud document`table` de l'élément, du nom local donné et de l' espace de noms HTML.

L' `caption` attribut IDL doit retourner, lors de l'obtention, le premier captionélément enfant de l' tableélément, s'il y en a un, ou null sinon. Lors de la définition, le premier caption élément enfant de l' tableélément, le cas échéant, doit être supprimé et la nouvelle valeur, si elle n'est pas nulle, doit être insérée en tant que premier nœud de l' tableélément.

La `createCaption()` méthode doit renvoyer le premier captionélément enfant de l' tableélément, le cas échéant ; sinon, un nouvel captionélément doit être table-created, inséré en tant que premier nœud de l' tableélément, puis renvoyé.

La `deleteCaption()` méthode doit supprimer le premier captionélément enfant de l' tableélément, le cas échéant.

L' `thead`attribut IDL doit retourner, lors de l'obtention, le premier theadélément enfant de l' tableélément, s'il y en a un, ou null sinon. Lors de la définition, si la nouvelle valeur est nulle ou un theadélément, le premier theadélément enfant de l' table élément, le cas échéant, doit être supprimé, et la nouvelle valeur, si elle n'est pas nulle, doit être insérée immédiatement avant le premier élément de l' tableélément qui est ni un caption élément ni un colgroupélément, le cas échéant, ou à la fin du tableau s'il n'y a pas de tels éléments. Si la nouvelle valeur n'est ni nulle ni un theadélément, alors un `"HierarchyRequestError" DOMException` doit être lancé à la place.

La `createTHead()` méthode doit renvoyer le premier theadélément enfant de l' tableélément, le cas échéant ; sinon, un nouvel theadélément doit être créé dans le tableau et inséré immédiatement avant le premier élément de l' tableélément qui n'est ni un captionélément ni un colgroupélément, le cas échéant, ou à la fin du tableau s'il n'y a pas de tels éléments, puis ce nouvel élément doit être retourné.

La `deleteTHead()` méthode doit supprimer le premier theadélément enfant de l' tableélément, le cas échéant.

L' `tFoot`attribut IDL doit retourner, lors de l'obtention, le premier tfootélément enfant de l' tableélément, s'il y en a un, ou null sinon. Lors de la définition, si la nouvelle valeur est nulle ou un tfootélément, le premier tfootélément enfant de l' table élément, le cas échéant, doit être supprimé, et la nouvelle valeur, si elle n'est pas nulle, doit être insérée à la fin du tableau. Si la nouvelle valeur n'est ni nulle ni un tfootélément, alors un "`HierarchyRequestError`" `DOMException` doit être lancé à la place.

La `createTFoot()` méthode doit renvoyer le premier tfootélément enfant de l' tableélément, le cas échéant ; sinon, un nouvel tfootélément doit être créé dans la table et inséré à la fin de la table, puis ce nouvel élément doit être renvoyé.

La `deleteTFoot()` méthode doit supprimer le premier tfootélément enfant de l' tableélément, le cas échéant.

L' `tBodies` attribut doit renvoyer un HTMLCollectionenraciné au tablenœud, dont le filtre correspond uniquement tbodyaux éléments enfants de l' table élément.

La `createTBody()` méthode doit table-créer un nouvel tbodyélément, l'insérer immédiatement après le dernier tbodyélément enfant dans l' tableélément, le cas échéant, ou à la fin de l' tableélément si l' tableélément n'a pas tbodyd'élément enfant, puis doit renvoyer le nouvel tbodyélément.

L' `rows` attribut doit renvoyer un HTMLCollectionenraciné au tablenœud, dont le filtre correspond uniquement traux éléments qui sont soit des enfants de l' table élément, soit des enfants de thead, tbody, ou tfootdes éléments qui sont eux-mêmes des enfants de l' tableélément. Les éléments de la collection doivent être ordonnés de telle sorte que les éléments dont le parent est a theadsoient inclus en premier, dans l'ordre de l'arborescence , suivis des éléments dont le parent est soit un élément table soit a tbody, toujours dans l' ordre de l'arborescence , suivis enfin des éléments dont le parent est un tfootélément, toujours dans l'ordre de l'arborescence .

Le comportement de la méthode dépend de l'état de la table. Lorsqu'elle est appelée, la méthode doit agir comme requis par le premier élément de la liste de conditions suivante qui décrit l'état de la table et l' argument *d'index* :`insertRow(index)`

**Si *index* est inférieur à -1 ou supérieur au nombre d'éléments dans rowsla collection :**

La méthode doit lancer un "`IndexSizeError`" `DOMException` .

**Si la rowscollection ne contient aucun élément et que la collection n'en tablecontient aucun tbody :**

La méthode doit [table-créer](#) un [tbody](#) élément, puis [table-créer](#) un [tr](#)élément, puis ajouter l' [tr](#)élément à l' [tbody](#)élément, puis ajouter l' [tbody](#)élément à l' [table](#)élément, et enfin retourner l' [tr](#)élément.

**Si la [rows](#)collection ne contient aucun élément :**

La méthode doit [table-créer](#) un [tr](#)élément, l'ajouter au dernier [tbody](#)élément de la table et renvoyer l' [tr](#) élément.

**Si *index* est -1 ou égal au nombre d'éléments dans [rows](#)la collection :**

La méthode doit [table-créer](#) un [tr](#)élément et l'ajouter au parent du dernier [tr](#)élément de la [rows](#)collection. Ensuite, l' [tr](#)élément nouvellement créé doit être renvoyé.

**Sinon:**

La méthode doit [table-créer](#) un [tr](#)élément, l'insérer immédiatement avant l' *index* ème [tr](#)élément dans la collection, dans le même parent, et enfin doit retourner l' élément [rows](#)nouvellement créé [.tr](#)

Lorsque la méthode est appelée, l'agent utilisateur doit exécuter les étapes suivantes :[deleteRow\(index\)](#)

1. Si *index* est inférieur à -1 ou supérieur ou égal au nombre d'éléments de la [rows](#)collection, lancez un "[IndexSizeError](#)" [DOMException](#) .
2. Si *index* est -1, [supprimez](#) le dernier élément de la [rows](#)collection de son parent, ou ne faites rien si la [rows](#)collection est vide.
3. Sinon, [supprimez](#) l' *index* ème élément de la [rows](#)collection de son parent.

Voici un exemple de tableau utilisé pour annoter un puzzle Sudoku. Observez l'absence d'en-têtes, qui ne sont pas nécessaires dans un tel tableau.

```
<style>
#sudoku { border-collapse: collapse; border: solid thick; }
#sudoku colgroup, table#sudoku tbody { border: solid medium; }
#sudoku td { border: solid thin; height: 1.4em; width: 1.4em;
text-align: center; padding: 0; }
</style>
<h1>Today's Sudoku</h1>
<table id="sudoku">
  <colgroup><col><col><col>
  <colgroup><col><col><col>
  <colgroup><col><col><col>
  <tbody>
    <tr> <td> 1 <td>    <td> 3 <td> 6 <td>    <td> 4 <td> 7 <td>    <td>
9
    <tr> <td>    <td> 2 <td>    <td>    <td> 9 <td>    <td>    <td> 1 <td>
```

```

        <tr> <td> 7 <td>      <td>      <td>      <td>      <td>      <td>      <td>
6
    <tbody>
        <tr> <td> 2 <td>      <td> 4 <td>      <td> 3 <td>      <td> 9 <td>      <td>
8
        <tr> <td>      <td>      <td>      <td>      <td>      <td>      <td>      <td>
        <tr> <td> 5 <td>      <td>      <td> 9 <td>      <td> 7 <td>      <td>      <td>
1
    <tbody>
        <tr> <td> 6 <td>      <td>      <td>      <td> 5 <td>      <td>      <td>      <td>
2
        <tr> <td>      <td>      <td>      <td>      <td> 7 <td>      <td>      <td>      <td>
        <tr> <td> 9 <td>      <td>      <td> 8 <td>      <td> 2 <td>      <td>      <td>
5
    </tbody>
</table>

```

#### 4.9.1.1 Techniques de description des tableaux

Pour les tableaux qui consistent en plus qu'une simple grille de cellules avec des en-têtes dans la première ligne et des en-têtes dans la première colonne, et pour tout tableau en général où le lecteur pourrait avoir des difficultés à comprendre le contenu, les auteurs doivent inclure des informations explicatives présentant le tableau. Ces informations sont utiles pour tous les utilisateurs, mais sont particulièrement utiles pour les utilisateurs qui ne peuvent pas voir le tableau, par exemple les utilisateurs de lecteurs d'écran.

Ces informations explicatives doivent présenter l'objectif du tableau, décrire sa structure cellulaire de base, mettre en évidence les tendances ou les modèles et, de manière générale, enseigner à l'utilisateur comment utiliser le tableau.

Par exemple, le tableau suivant :

Caractéristiques avec des côtés positifs et négatifs

Négatif	Caractéristique	Positif
Triste	Humeur	Content
Échouer	Grade	Qui passe

... pourrait bénéficier d'une description expliquant la façon dont le tableau est présenté, quelque chose comme "Les caractéristiques sont données dans la deuxième colonne, avec le côté négatif dans la colonne de gauche et le côté positif dans la colonne de droite".

Il existe plusieurs façons d'inclure ces informations, telles que :

## En prose, entourant la table

```
<p>In the following table, characteristics are given in the
second
column, with the negative side in the left column and the
positive
side in the right column.</p>
<table>
  <caption>Characteristics with positive and negative
sides</caption>
  <thead>
    <tr>
      <th id="n"> Negative
      <th> Characteristic
      <th> Positive
    </tr>
  <tbody>
    <tr>
      <td headers="n r1"> Sad
      <th id="r1"> Mood
      <td> Happy
    </tr>
    <tr>
      <td headers="n r2"> Failing
      <th id="r2"> Grade
      <td> Passing
    </tr>
  </tbody>
</table>
```

## Dans le tableau caption

```
<table>
  <caption>
    <strong>Characteristics with positive and negative
sides.</strong>
    <p>Characteristics are given in the second column, with the
negative side in the left column and the positive side in
the right
column.</p>
  </caption>
  <thead>
    <tr>
      <th id="n"> Negative
      <th> Characteristic
      <th> Positive
    </tr>
  <tbody>
    <tr>
      <td headers="n r1"> Sad
      <th id="r1"> Mood
      <td> Happy
    </tr>
    <tr>
      <td headers="n r2"> Failing
      <th id="r2"> Grade
      <td> Passing
    </tr>
  </tbody>
</table>
```

## Dans le tableau caption, dans un details élément

```
<table>
  <caption>
    <strong>Characteristics with positive and negative
sides.</strong>
    <details>
      <summary>Help</summary>
      <p>Characteristics are given in the second column, with
the
```

```

        negative side in the left column and the positive side in
the right
        column.</p>
    </details>
</caption>
<thead>
<tr>
    <th id="n"> Negative
    <th> Characteristic
    <th> Positive
<tbody>
<tr>
    <td headers="n r1"> Sad
    <th id="r1"> Mood
    <td> Happy
<tr>
    <td headers="n r2"> Failing
    <th id="r2"> Grade
    <td> Passing
</table>

```

A côté de la table, dans le même figure

```

<figure>
    <figcaption>Characteristics with positive and negative
sides</figcaption>
    <p>Characteristics are given in the second column, with the
negative side in the left column and the positive side in
the right
    column.</p>
    <table>
    <thead>
    <tr>
        <th id="n"> Negative
        <th> Characteristic
        <th> Positive
    <tbody>
    <tr>
        <td headers="n r1"> Sad
        <th id="r1"> Mood
        <td> Happy
    <tr>
        <td headers="n r2"> Failing
        <th id="r2"> Grade
        <td> Passing
    </table>
</figure>

```

A côté de la table, dans un figures figcaption

```

<figure>
    <figcaption>
        <strong>Characteristics with positive and negative
sides</strong>
        <p>Characteristics are given in the second column, with the
negative side in the left column and the positive side in
the right
        column.</p>
    </figcaption>
    <table>
    <thead>
    <tr>
        <th id="n"> Negative

```

```

    <th> Characteristic
    <th> Positive
  <tbody>
    <tr>
      <td headers="n r1"> Sad
      <th id="r1"> Mood
      <td> Happy
    <tr>
      <td headers="n r2"> Failing
      <th id="r2"> Grade
      <td> Passing
  </table>
</figure>

```

Les auteurs peuvent également utiliser d'autres techniques, ou des combinaisons des techniques ci-dessus, selon le cas.

La meilleure option, bien sûr, plutôt que d'écrire une description expliquant la façon dont le tableau est disposé, est d'ajuster le tableau de sorte qu'aucune explication ne soit nécessaire.

Dans le cas du tableau utilisé dans les exemples ci-dessus, un simple réarrangement du tableau de sorte que les en-têtes soient en haut et à gauche supprime le besoin d'une explication ainsi que le besoin d'utiliser des attributs headers :

```

<table>
  <caption>Characteristics with positive and negative
  sides</caption>
  <thead>
    <tr>
      <th> Characteristic
      <th> Negative
      <th> Positive
    <tbody>
      <tr>
        <th> Mood
        <td> Sad
        <td> Happy
      <tr>
        <th> Grade
        <td> Failing
        <td> Passing
    </table>

```

#### 4.9.1.2 Techniques de conception de table



Une bonne conception de table est essentielle pour rendre les tables plus lisibles et utilisables.

Dans les médias visuels, fournir des bordures de colonnes et de lignes et des arrière-plans de lignes alternés peut être très efficace pour rendre les tableaux complexes plus lisibles.

Pour les tableaux avec de gros volumes de contenu numérique, l'utilisation de polices à espacement fixe peut aider les utilisateurs à voir les modèles, en particulier dans les situations où un agent utilisateur n'affiche pas les bordures. (Malheureusement, pour des raisons historiques, ne pas afficher les bordures sur les tables est une valeur par défaut courante.)

Dans les médias vocaux, les cellules de tableau peuvent être distinguées en signalant les en-têtes correspondants avant de lire le contenu de la cellule, et en permettant aux utilisateurs de naviguer dans le tableau sous forme de grille, plutôt que de sérialiser l'intégralité du contenu du tableau dans l'ordre des sources.

Les auteurs sont encouragés à utiliser CSS pour obtenir ces effets.

Les agents utilisateurs sont encouragés à rendre les tableaux en utilisant ces techniques chaque fois que la page n'utilise pas CSS et que le tableau n'est pas classé comme tableau de mise en page.

#### 4.9.2 L' **caption** élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

En tant que premier élément enfant d'un [table](#) élément.

##### Modèle de contenu :

[Contenu du flux](#) , mais sans [table](#) éléments descendants.

##### Omission de balise dans text/html :

La [balise de fin](#) [caption](#) d'un élément peut être omise si l' élément n'est pas immédiatement suivi d' [un espace ASCII](#) ou d'un [commentaire](#) [.caption](#)

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

## Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLTableCaptionElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  // also has obsolete members
```

```
};
```

L' [caption](#)élément [représente](#) le titre du [table](#) qui est son parent, s'il a un parent et qui est un [table](#)élément.

L' [caption](#)élément participe au [modèle de table](#) .

Lorsqu'un [table](#)élément est le seul contenu d'un [figure](#)élément autre que le [figcaption](#), l' [caption](#)élément doit être omis au profit du [figcaption](#).

Une légende peut introduire le contexte d'un tableau, ce qui le rend beaucoup plus facile à comprendre.

Considérons, par exemple, le tableau suivant :

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	dix
5	6	7	8	9	dix	11
6	7	8	9	dix	11	12

Dans l'abstrait, ce tableau n'est pas clair. Cependant, avec une légende donnant le numéro du tableau (pour [référence](#) dans la prose principale) et expliquant son utilisation, cela a plus de sens :

```
<caption>
```

```
<p>Table 1.
```

```
<p>This table shows the total score obtained from rolling two  
six-sided dice. The first row represents the value of the first  
die,
```

```
the first column the value of the second die. The total is given in  
the cell that corresponds to the values of the two dice.
```

</caption>

Cela fournit à l'utilisateur plus de contexte :

Tableau 1.

Ce tableau montre le score total obtenu en lançant deux dés à six faces. La première ligne représente la valeur du premier dé, la première colonne la valeur du second dé. Le total est donné dans la cellule qui correspond aux valeurs des deux dés.

	1	2	3	4	5	6
1	2	3	4	5	6	7
2	3	4	5	6	7	8
3	4	5	6	7	8	9
4	5	6	7	8	9	dix
5	6	7	8	9	dix	11
6	7	8	9	dix	11	12

#### 4.9.3 L' `colgroup` élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un `table` élément, après tous `caption` les éléments et avant tous les éléments `thead`, `tbody`, `tfoot` et `tr` .

##### Modèle de contenu :

Si l' `span` attribut est présent : `Nothing` .

Si l' `span` attribut est absent : Zéro ou plus `col` et `template` éléments.

##### Omission de balise dans text/html :

La `balise de débutcolgroup` d'un élément peut être omise si la première chose à l'intérieur de l' élément est un élément, et si l'élément n'est pas immédiatement précédé d'un autre élément dont `la balise de fin` a été omise. (Il ne peut pas être omis si l'élément est vide.)`colgroupcolcolgroup`

La `balise de fincolgroup` d'un élément peut être omise si l' élément n'est pas immédiatement suivi d' `un espace ASCII` ou d'un `commentaire` .`colgroup`

##### Attributs de contenu :

Attributs globaux

`span` — Nombre de colonnes couvertes par l'élément

## Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

## Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLTableColElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute unsigned long span;
```

```
    // also has obsolete members
```

```
};
```

L' colgroupélément représente un groupe d'une ou plusieurs colonnes dans table qui est son parent, s'il a un parent et qui est un tableélément.

Si l' colgroupélément ne contient aucun colélément, alors l'élément peut avoir un span attribut de contenu spécifié, dont la valeur doit être un entier non négatif valide supérieur à zéro et inférieur ou égal à 1 000.

L' colgroupélément et son span attribut participent au modèle de table .

L' span attribut IDL doit refléter l'attribut de contenu du même nom. Il est limité à la plage [1, 1000] et sa valeur par défaut est 1.

### 4.9.4 L' colélément



## Catégories :

Aucun.

## Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un colgroupélément qui n'a pas d' spanattribut.

## Modèle de contenu :

Rien .

### Omission de balise dans text/html :

Pas de balise de fin .

### Attributs de contenu :

Attributs globaux

span— Nombre de colonnes couvertes par l'élément

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

Utilise HTMLTableColElement, tel que défini pour colgroup les éléments.

Si un colélément a un parent et qu'il s'agit d'un colgroupélément qui a lui-même un parent qui est un tableélément, alors l' colélément représente une ou plusieurs colonnes dans le groupe de colonnes représenté par that colgroup.

L'élément peut avoir un spanattribut de contenu spécifié, dont la valeur doit être un entier non négatif valide supérieur à zéro et inférieur ou égal à 1 000.

L' colélément et son spanattribut participent au modèle de table .

## 4.9.5 L' tbodyélément



### Catégories :

Aucun.

### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un tableélément, après tous les éléments caption, colgroupet thead, mais uniquement s'il n'existe aucun trélément enfant de l' tableélément.

### Modèle de contenu :

Zéro ou plus tret éléments de support de script .

### Omission de balise dans text/html :

La balise de débuttbody d'un élément peut être omise si la première chose à l'intérieur de l' élément est un élément, et si l'élément n'est pas immédiatement précédé d'un élément , ou dont la balise de fin a été omise. (Il ne peut pas être omis si l'élément est vide.)tbodytrtbodytheadtfoot

La balise de fin `tbody` d'un élément peut être omise si l'élément est immédiatement suivi d'un élément ou s'il n'y a plus de contenu dans l'élément parent. `tbodytbodytfoot`

### Attributs de contenu :

Attributs globaux

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLTableSectionElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [SameObject] readonly attribute HTMLCollection rows;
```

```
  HTMLTableRowElement insertRow(optional long index = -1);
```

```
  [CEReactions] undefined deleteRow(long index);
```

```
  // also has obsolete members
```

```
};
```

L' [HTMLTableSectionElement](#) interface est également utilisée pour les éléments `thead` et `tfoot`.

L' `tbody` élément représente un bloc de lignes constitué d'un corps de données pour l' `table` élément parent, si l' `tbody` élément a un parent et qu'il s'agit d'un fichier `table`.

L' `tbody` élément participe au modèle de table .

#### `tbody.rows`

Renvoie un [HTMLCollection](#) des `tr` éléments de la section table.

```
tr = tbody.insertRow([ index ])
```

Crée un `tr` élément, l'insère dans la section table à la position donnée par l'argument et renvoie le `tr`.

La position est relative aux lignes dans la section de tableau. L'indice `-1`, qui est la valeur par défaut si l'argument est omis, équivaut à insérer à la fin de la section table.

Si la position donnée est inférieure à `-1` ou supérieure au nombre de lignes, lance un `"IndexSizeError" DOMException`.

**`tbody.deleteRow(index)`**

Supprime l'`tr`élément avec la position donnée dans la section table.

La position est relative aux lignes dans la section de tableau. L'indice `-1` équivaut à supprimer la dernière ligne de la section de table.

Si la position donnée est inférieure à `-1` ou supérieure à l'index de la dernière ligne, ou s'il n'y a pas de ligne, lance un `"IndexSizeError" DOMException`.

L' `rows` attribut doit renvoyer une `HTMLCollection` racine à cet élément, dont le filtre correspond uniquement `tr`aux éléments enfants de cet élément.

La méthode doit agir comme suit : `insertRow(index)`

1. Si `index` est inférieur à `-1` ou supérieur au nombre d'éléments de la `rows` collection, lancez un `"IndexSizeError" DOMException`.
2. Soit `table row` le résultat de [la création d'un élément](#) étant donné le `nœud document`, `tr` et l' [espace de noms HTML](#) de cet élément.
3. Si `index` est `-1` ou égal au nombre d'éléments dans la `rows` collection, alors [ajoutez une ligne de table](#) à cet élément.
4. Sinon, [insérez la ligne du tableau](#) en tant qu'enfant de cet élément, immédiatement avant l' `index` ème `tr`élément dans la `rows` collection.
5. Renvoie *la ligne du tableau*.

La méthode doit, lorsqu'elle est invoquée, agir comme suit : `deleteRow(index)`

1. Si `index` est inférieur à `-1` ou supérieur ou égal au nombre d'éléments de la `rows` collection, lancez un `"IndexSizeError" DOMException`.
2. Si `index` est `-1`, [supprimez](#) le dernier élément de la `rows` collection de cet élément, ou ne faites rien si la `rows` collection est vide.
3. Sinon, [supprimez](#) l' `index` ème élément de la `rows` collection de cet élément.

#### 4.9.6 L' `thead`élément

### Catégories :

Aucun.

### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un [table](#) élément, après tous les éléments [caption](#), et [colgroup](#) et avant tous les éléments [tbody](#), [tfoot](#) et [tr](#), mais uniquement s'il n'y a pas d'autres [thead](#) éléments enfants de l' [table](#) élément.

### Modèle de contenu :

Zéro ou plus [tr](#) et éléments [de support de script](#) .

### Omission de balise dans text/html :

La [balise de fin](#) [thead](#) d'un élément peut être omise si l' élément est immédiatement suivi d'un élément ou [.theadtbodytfoot](#)

### Attributs de contenu :

[Attributs globaux](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### Interface DOM :

Utilise [HTMLTableSectionElement](#), tel que défini pour [tbody](#) les éléments.

L' [thead](#) élément [représente](#) le [bloc](#) de [lignes](#) constitué des étiquettes de colonne (en-têtes) pour l' [table](#) élément parent, si l' [thead](#) élément a un parent et s'il s'agit d'un fichier [table](#).

L' [thead](#) élément participe au [modèle de table](#) .

Cet exemple montre un [thead](#) élément utilisé. Remarquez l'utilisation des éléments [th](#) et [td](#) dans l' [thead](#) élément : la première ligne correspond aux en-têtes et la deuxième ligne explique comment remplir le tableau.

```
<table>
  <caption> School auction sign-up sheet </caption>
  <thead>
    <tr>
      <th><label for=e1>Name</label>
      <th><label for=e2>Product</label>
      <th><label for=e3>Picture</label>
      <th><label for=e4>Price</label>
    <tr>
      <td>Your name here
      <td>What are you selling?
      <td>Link to a picture
```



```

<td>Your reserve price
</td>
</tr>
<tr>
<td>Ms Danus
<td>Doughnuts
<td>
<td>$45
</tr>
<tr>
<td><input id=e1 type=text name=who required form=f>
<td><input id=e2 type=text name=what required form=f>
<td><input id=e3 type=url name=pic form=f>
<td><input id=e4 type=number step=0.01 min=0 value=0 required
form=f>
</tr>
</table>
<form id=f action="/auction.cgi">
<input type=button name=add value="Submit">
</form>

```

#### 4.9.7 L' **tfoot** élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un [table](#) élément, après tous les éléments [caption](#), [colgroup](#), [thead](#), [tbody](#) et [tr](#), mais uniquement s'il n'existe aucun autre [tfoot](#) élément enfant de l' [table](#) élément.

##### Modèle de contenu :

Zéro ou plus [tr](#) et éléments [de support de script](#).

##### Omission de balise dans text/html :

La [balise de fin tfoot](#) d'un élément peut être omise s'il n'y a plus de contenu dans l'élément parent.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

## Interface DOM :

Utilise [HTMLTableSectionElement](#), tel que défini pour [tbody](#) les éléments.

L' [tfoot](#) élément [représente](#) le [bloc](#) de [lignes](#) constitué des résumés de colonne (pieds de page) pour l' [table](#) élément parent, si l' [tfoot](#) élément a un parent et s'il s'agit d'un fichier [table](#).

L' [tfoot](#) élément participe au [modèle de table](#) .

### 4.9.8 L' [tr](#) élément



## Catégories :

Aucun.

## Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un [thead](#) élément.  
En tant qu'enfant d'un [tbody](#) élément.  
En tant qu'enfant d'un [tfoot](#) élément.  
En tant qu'enfant d'un [table](#) élément, après tous les éléments [caption](#), [colgroup](#) et [thead](#) , mais uniquement s'il n'existe aucun [tbody](#) élément enfant de l' [table](#) élément.

## Modèle de contenu :

Zéro ou plusieurs éléments [td](#), [th](#) et [de support de script](#) .

## Omission de balise dans text/html :

La [balise de fin](#) [tr](#) d'un élément peut être omise si l' élément est immédiatement suivi d'un autre élément ou s'il n'y a plus de contenu dans l'élément parent. [trtr](#)

## Attributs de contenu :

[Attributs globaux](#)

## Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

## Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLTableRowElement : HTMLElement {
```

```

[HTMLConstructor] constructor();

readonly attribute long rowIndex;

readonly attribute long sectionRowIndex;

[SameObject] readonly attribute HTMLCollection cells;

HTMLTableCellElement insertCell(optional long index = -
1);

[CEReactions] undefined deleteCell(long index);

// also has obsolete members

};

```

L' trélément représente une ligne de cellules dans un tableau .

L' trélément participe au modèle de table .

#### tr.rowIndex

✓ 

Renvoie la position de la ligne dans la rows liste du tableau.

Renvoie -1 si l'élément n'est pas dans un tableau.

#### tr.sectionRowIndex

Renvoie la position de la ligne dans la rows liste de la section table.

Renvoie -1 si l'élément n'est pas dans une section de tableau.

#### tr.cells

Renvoie un HTMLCollection des éléments td et th de la ligne.

`cell = tr.insertCell([ index ])`

✓ 

Crée un td élément, l'insère dans la ligne du tableau à la position donnée par l'argument et renvoie le td.

La position est relative aux cellules de la ligne. L'indice -1, qui est la valeur par défaut si l'argument est omis, équivaut à insérer à la fin de la ligne.

Si la position donnée est inférieure à -1 ou supérieure au nombre de cellules, lance un "IndexSizeError" DOMException .

**`tr.deleteCell(index)`**

Supprime l'élément `td` ou `th` avec la position donnée dans la ligne.

La position est relative aux cellules de la ligne. L'indice -1 équivaut à supprimer la dernière cellule de la ligne.

Si la position donnée est inférieure à -1 ou supérieure à l'index de la dernière cellule, ou s'il n'y a pas de cellule, lance un `"IndexSizeError" DOMException`.

L' `rowIndex` attribut doit, si cet élément a un `table` élément parent, ou un élément parent `tbody`, `thead`, ou `tfoot` et un élément *grand-parent* `table`, retourner l'index de cet `tr` élément dans la collection `table` de cet élément `rows`. S'il n'y a pas un tel `table` élément, alors l'attribut doit retourner -1.

L' `sectionRowIndex` attribut doit, si cet élément a un élément parent `table`, `tbody`, `thead` ou `tfoot`, renvoyer l'index de l' `tr` élément dans la collection de l'élément parent `rows` (pour les tables, c'est `HTMLTableElement` la `rows` collection ; pour les sections de table, c'est `HTMLTableSectionElement` la `rows` collection). S'il n'y a pas un tel élément parent, alors l'attribut doit retourner -1.

L' `cells` attribut doit renvoyer une `HTMLCollection` racine à cet `tr` élément, dont le filtre correspond uniquement `td` et `th` les éléments qui sont des enfants de l' `tr` élément.

La méthode doit agir comme suit : `insertCell(index)`

1. Si `index` est inférieur à -1 ou supérieur au nombre d'éléments de la `cells` collection, lancez un `"IndexSizeError" DOMException`.
2. Supposons que `table cell` soit le résultat de [la création d'un élément](#) étant donné le [nœud document](#), et l' [espace de noms HTML](#) `tr` de cet élément `td`.
3. Si `index` est égal à -1 ou égal au nombre d'éléments dans `cells` la collection, alors [ajoutez la cellule du tableau](#) à cet `tr` élément.
4. Sinon, [insérez la cellule du tableau](#) en tant qu'enfant de cet `tr` élément, immédiatement avant l' `index` e `td` ou `th` l'élément dans la `cells` collection.
5. *Cellule du tableau* de retour.

La méthode doit agir comme suit : `deleteCell(index)`

1. Si `index` est inférieur à -1 ou supérieur ou égal au nombre d'éléments de la `cells` collection, lancez un `"IndexSizeError" DOMException`.
2. Si `index` est -1, [supprimez](#) le dernier élément de la `cells` collection de son parent, ou ne faites rien si la `cells` collection est vide.

- Sinon, [supprimez](#) l' *index* ème élément de la [cells](#) collection de son parent.

#### 4.9.9 L' [td](#) élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un [tr](#) élément.

##### Modèle de contenu :

[Contenu du flux](#) .

##### Omission de balise dans text/html :

La [balise de fin](#) [td](#) d'un élément peut être omise si l' élément est immédiatement suivi d'un élément ou s'il n'y a plus de contenu dans l'élément parent. [tdtdth](#)

##### Attributs de contenu :

[Attributs globaux](#)

[colspan](#)— Nombre de colonnes sur lesquelles la cellule doit s'étendre

[rowspan](#)— Nombre de lignes sur lesquelles la cellule doit s'étendre

[headers](#)— Les cellules d'en-tête de cette cellule

##### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

##### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLTableCellElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute unsigned long colSpan;
```

```
    [CEReactions] attribute unsigned long rowSpan;
```

```
    [CEReactions] attribute DOMString headers;
```

```

readonly attribute long cellIndex;

[CEReactions] attribute DOMString scope; // only
conforming for th elements

[CEReactions] attribute DOMString abbr; // only
conforming for th elements

// also has obsolete members

};

```

L' [HTMLTableCellElement](#) interface est également utilisée pour [th](#) les éléments.

L' [td](#) élément [représente](#) une [cellule](#) de données dans un tableau.

L' [td](#) élément et ses attributs [colspan](#), [rowspan](#) et [headers](#) participent au [modèle de table](#) .

Les agents utilisateurs, en particulier dans les environnements non visuels ou lorsque l'affichage du tableau sous forme de grille 2D n'est pas pratique, peuvent donner à l'utilisateur le contexte de la cellule lors du rendu du contenu d'une cellule ; par exemple, donner sa position dans le [modèle de table](#) , ou lister les cellules d'en-tête de la cellule (telle que déterminée par l' [algorithme d'attribution des cellules d'en-tête](#) ). Lorsque les cellules d'en-tête d'une cellule sont listées, les agents utilisateurs peuvent utiliser la valeur des [abbr](#) attributs de ces cellules d'en-tête, s'il y en a, au lieu du contenu des cellules d'en-tête elles-mêmes.

Dans cet exemple, nous voyons un extrait d'une application Web consistant en une grille de cellules modifiables (essentiellement une simple feuille de calcul). L'une des cellules a été configurée pour afficher la somme des cellules au-dessus. Trois ont été marqués comme titres, qui utilisent [th](#) des éléments au lieu d' [td](#) éléments. Un script attacherait des gestionnaires d'événements à ces éléments pour maintenir le total.

```

<table>
  <tr>
    <th><input value="Name">
    <th><input value="Paid ($)">
  <tr>
    <td><input value="Jeff">
    <td><input value="14">
  <tr>

```

```
 <input value="Britta">  <input value="9"> <input value="Abed">  <input value="25"> <input value="Shirley">  <input value="2"> <input value="Annie">  <input value="5"> <input value="Troy">  <input value="5"> <input value="Pierce">  <input value="1000"> <input value="Total">  <output value="1060"> </table> | | | | | | | |
```

#### 4.9.10 L' **th**élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un **tr**élément.

##### Modèle de contenu :

Contenu de flux , mais sans header, footer, contenu de sectionnement ou descendants de contenu d'en-tête .

##### Omission de balise dans text/html :

La balise de fin**th** d'un élément peut être omise si l' élément est immédiatement suivi d'un élément ou s'il n'y a plus de contenu dans l'élément parent.**thtdth**

##### Attributs de contenu :

Attributs globaux

colspan— Nombre de colonnes sur lesquelles la cellule doit s'étendre  
rowspan— Nombre de lignes sur lesquelles la cellule doit s'étendre  
headers— Les cellules d'en-tête de cette cellule  
scope— Spécifie les cellules auxquelles la cellule d'en-tête s'applique  
abbr— Libellé alternatif à utiliser pour la cellule d'en-tête lors du référencement de la cellule dans d'autres contextes

### **Considérations d'accessibilité :**

Pour les auteurs .  
Pour les exécutants .

### **Interface DOM :**

Utilise HTMLTableCellElement, tel que défini pour td les éléments.

L' thélément représente une cellule d'en-tête dans un tableau.

L' thélément peut avoir un scopeattribut de contenu spécifié. L' scopeattribut est un attribut énuméré avec cinq états, dont quatre ont des mots clés explicites :

#### **Le rowmot-clé, qui correspond à l' état de la *ligne***

L' état de la *ligne* signifie que la cellule d'en-tête s'applique à certaines des cellules suivantes dans la ou les mêmes lignes.

#### **Le colmot-clé, qui correspond à l' état de la *colonne***

L' état de la *colonne* signifie que la cellule d'en-tête s'applique à certaines des cellules suivantes dans la ou les mêmes colonnes.

#### **Le rowgroupmot-clé, qui correspond à l' état du *groupe de lignes***

L' état du *groupe de lignes* signifie que la cellule d'en-tête s'applique à toutes les cellules restantes du groupe de lignes. L'attribut d' un thélément scopene doit pas être dans l' état groupe de lignes si l'élément n'est pas ancré dans un groupe de lignes .

#### **Le colgroupmot-clé, qui correspond à l' état du *groupe de colonnes***

L' état du *groupe de colonnes* signifie que la cellule d'en-tête s'applique à toutes les cellules restantes du groupe de colonnes. L' attribut thd'un élément scopene doit pas être dans l' état groupe de colonnes si l'élément n'est pas ancré dans un groupe de colonnes .

#### **L' état *automatique***

L' état *automatique* fait que la cellule d'en-tête s'applique à un ensemble de cellules sélectionnées en fonction du contexte.

La valeur par défaut manquante et la valeur par défaut non validescope de l'attribut correspondent à l' état *automatique* .

L' thélément peut avoir un abbrattribut de contenu spécifié. Sa valeur doit être une étiquette alternative pour la cellule d'en-tête, à utiliser lors du référencement de la cellule dans d'autres contextes (par exemple lors de la description des cellules d'en-



tête qui s'appliquent à une cellule de données). Il s'agit généralement d'une forme abrégée de la cellule d'en-tête complète, mais il peut également s'agir d'une extension ou simplement d'une formulation différente.

L' thélément et ses attributs colspan, rowspan, headers et scope participent au modèle de table .

L'exemple suivant montre comment la valeur scope de l'attribut rowgroup affecte les cellules de données auxquelles s'applique une cellule d'en-tête.

Voici un fragment de balisage montrant un tableau :

```
<table>
  <thead>
    <tr> <th> ID <th> Measurement <th> Average <th> Maximum
  <tbody>
    <tr> <td> <th scope=rowgroup> Cats <td> <td>
    <tr> <td> 93 <th> Legs <td> 3.5 <td> 4
    <tr> <td> 10 <th> Tails <td> 1 <td> 1
  <tbody>
    <tr> <td> <th scope=rowgroup> English speakers <td> <td>
    <tr> <td> 32 <th> Legs <td> 2.67 <td> 4
    <tr> <td> 35 <th> Tails <td> 0.33 <td> 1
</table>
```

Cela donnerait le tableau suivant :

IDENTIFIANT	La mesure	Moyenne	Maximum
	Chats		
93	Jambes	3.5	4
dix	Queues	1	1
	anglophones		
32	Jambes	2,67	4
35	Queues	0,33	1

Les en-têtes de la première ligne s'appliquent tous directement aux lignes de leur colonne.

Les en-têtes avec un scope attribut dans l' état rowgroup s'appliquent à toutes les cellules de leur groupe de lignes autres que les cellules de la première colonne.

Les en-têtes restants s'appliquent uniquement aux cellules situées à leur droite.

ID	Measurement	Average	Maximum
	Cats		
93	Legs	3.5	4
10	Tails	1	1
	English speakers		
32	Legs	2.67	4
35	Tails	0.33	1

#### 4.9.11 Attributs communs aux éléments td et th

Les éléments td et th peuvent avoir un **colspan** attribut de contenu spécifié, dont la valeur doit être un [entier non négatif valide](#) supérieur à zéro et inférieur ou égal à 1 000.

Les éléments td et th peuvent également avoir un **rowspan** attribut de contenu spécifié, dont la valeur doit être un [entier non négatif valide](#) inférieur ou égal à 65534. Pour cet attribut, la valeur zéro signifie que la cellule doit s'étendre sur toutes les lignes restantes du groupe de lignes. .

Ces attributs donnent respectivement le nombre de colonnes et de lignes que la cellule doit couvrir. Ces attributs ne doivent pas être utilisés pour superposer des cellules, comme décrit dans la description du [modèle de table](#) .

L' élément td et th peut avoir un **headers** attribut de contenu spécifié. L' **headers** attribut, s'il est spécifié, doit contenir une chaîne constituée d'un [ensemble non ordonné de jetons uniques séparés par des espaces](#) , dont aucun n'est [identique à](#) un autre jeton et dont chacun doit avoir la valeur d'un ID d'un th élément participant à la même [table](#) en tant qu'élément td ou th (tel que défini par le [modèle de table](#) ).

Un th élément avec l'[identifiant](#) *id* est dit être *directement ciblé* par tous les éléments td et th dans la même [table](#) qui ont **headers** des attributs dont les valeurs incluent comme l'un de leurs jetons l' [identifiant](#) *id* . Un th élément *A* est dit *visé* par

a th ou par td l'élément *B* si soit *A* est *directement visé* par *B* soit s'il existe un élément *C* lui-même *visé* par l'élément *B* et *A* est *directement visé* par *C*.

Un th élément ne doit pas être *ciblé* par lui-même.

Les attributs colspan, rowspan et headers participent au [modèle de table](#).

---

#### **cell.cellIndex**

Renvoie la position de la cellule dans la cells liste de la ligne. Cela ne correspond pas nécessairement à la position *x* de la cellule dans le tableau, car les cellules précédentes peuvent couvrir plusieurs lignes ou colonnes.

Renvoie -1 si l'élément n'est pas dans une ligne.

L' **colSpan** attribut IDL doit [réfléter](#) l' colspan attribut content. Il est [limité à la plage](#) [1, 1000] et sa valeur par défaut est 1.

L' **rowSpan** attribut IDL doit [réfléter](#) l' rowspan attribut content. Il est [limité à la plage](#) [0, 65534] et sa valeur par défaut est 1.

L' **headers** attribut IDL doit [réfléter](#) l'attribut de contenu du même nom.

L' **cellIndex** attribut IDL doit, si l'élément a un tr élément parent, renvoyer l'index de l'élément de la cellule dans la cells collection de l'élément parent. S'il n'y a pas un tel élément parent, alors l'attribut doit retourner -1.

L' **scope** attribut IDL doit [réfléter](#) l'attribut de contenu du même nom, [limité aux seules valeurs connues](#).

L' **abbr** attribut IDL doit [réfléter](#) l'attribut de contenu du même nom.

### **4.9.12 Modèle de traitement**

Les divers éléments de table et leurs attributs de contenu définissent ensemble le **modèle de table**.

Un **tableau** est constitué de cellules alignées sur une grille bidimensionnelle de **cases** de coordonnées ( *x* , *y* ). La grille est finie et est soit vide, soit comporte un ou plusieurs emplacements. Si la grille a une ou plusieurs fentes, alors les coordonnées *x* sont toujours dans la plage  $0 \leq x < x_{width}$ , et les coordonnées *y* sont toujours dans la plage  $0 \leq y < y_{height}$ . Si l'une ou les deux de *la largeur x* et *de la*

$hauteur$   $y$  sont égales à zéro, alors la table est vide (n'a pas d'emplacements). Les tableaux correspondent à table éléments.

Une **cellule** est un ensemble d'emplacements ancrés à un emplacement ( $cellule_x$ ,  $cellule_y$ ), et avec une *largeur* et une *hauteur* particulières telles que la cellule couvre tous les emplacements de coordonnées ( $x$ ,  $y$ ) où  $cellule_x \leq x < cellule_x + largeur$  et  $cellule_y \leq y < cellule_y + hauteur$ . Les cellules peuvent être *des cellules de données* ou *des cellules d'en-tête*. Les cellules de données correspondent à td éléments et les cellules d'en-tête correspondent aux th éléments. Les cellules des deux types peuvent avoir zéro ou plusieurs cellules d'en-tête associées.

Il est possible, dans certains cas d'erreur, que deux cellules occupent le même emplacement.

Une **ligne** est un ensemble complet d'emplacements de  $x=0$  à  $x = x_{width} - 1$ , pour une valeur particulière de  $y$ . Les lignes correspondent généralement à tr des éléments, bien qu'un groupe de lignes puisse avoir des lignes implicites à la fin dans certains cas impliquant des cellules s'étendant sur plusieurs lignes.

Une **colonne** est un ensemble complet d'emplacements de  $y=0$  à  $y = y_{hauteur} - 1$ , pour une valeur particulière de  $x$ . Les colonnes peuvent correspondre à col des éléments. En l'absence d' col éléments, les colonnes sont implicites.

Un **groupe de lignes** est un ensemble de lignes ancrées à un emplacement ( $0$ ,  $groupe_y$ ) avec une *hauteur* particulière telle que le groupe de lignes couvre tous les emplacements de coordonnées ( $x$ ,  $y$ ) où  $0 \leq x < x_{largeur}$  et  $groupe_y \leq y < groupe_y + hauteur$ . Les groupes de lignes correspondent aux éléments tbody, thead et tfoot. Toutes les lignes ne sont pas nécessairement dans un groupe de lignes.

Un **groupe de colonnes** est un ensemble de colonnes ancrées à un emplacement ( $groupe_x$ ,  $0$ ) avec une *largeur* particulière telle que le groupe de colonnes couvre tous les emplacements de coordonnées ( $x$ ,  $y$ ) où  $groupe_x \leq x < groupe_x + largeur$  et  $0 \leq y < y_{hauteur}$ . Les groupes de colonnes correspondent aux colgroup éléments. Toutes les colonnes ne sont pas nécessairement dans un groupe de colonnes.

Les groupes de lignes ne peuvent pas se chevaucher. De même, les groupes de colonnes ne peuvent pas se chevaucher.

Une cellule ne peut pas couvrir les emplacements qui appartiennent à deux ou plusieurs groupes de lignes. Il est toutefois possible qu'une cellule se trouve dans plusieurs groupes de colonnes. Tous les emplacements qui font partie d'une cellule font partie de zéro ou d'un groupe de lignes et de zéro ou plusieurs groupes de colonnes.

Outre [les cellules](#) , [les colonnes](#) , [les lignes](#) , [les groupes de lignes](#) et [les groupes de colonnes](#) , [les tableaux](#) peuvent être [caption](#) associés à un élément. Cela donne au tableau un titre ou une légende.

Une **erreur de modèle de table** est une erreur avec les données représentées par [table](#) des éléments et leurs descendants. Les documents ne doivent pas contenir d'erreurs de modèle de table.

#### 4.9.12.1 Formation d'un tableau

Pour déterminer quels éléments correspondent à quels emplacements dans une [table](#) associée à un [table](#) élément, pour déterminer les dimensions de la table (  $x_{largeur}$  et  $y_{hauteur}$  ) et pour déterminer s'il y a des [erreurs de modèle de table](#) , les agents utilisateurs doivent utiliser l'algorithme suivant :

1. Soit  $x_{largeur}$  nulle.
2. Soit  $y_{hauteur}$  nulle.
3. Soit *les éléments en attente* [tfoot](#) une liste d' [tfoot](#) éléments, initialement vide.
4. Soit *la table* la [table](#) représentée par l' [table](#) élément. Les variables  $x_{largeur}$  et  $y_{hauteur}$  donnent les dimensions du *tableau* . *La table* est initialement vide.
5. Si l' [table](#) élément n'a pas d'éléments enfants, retournez *la table* (qui sera vide).
6. Associez le premier [caption](#) élément enfant de l' [table](#) élément à *la table* . S'il n'y a pas de tels enfants, alors il n'a pas [caption](#) d'élément associé.
7. Soit l' *élément courant* être le premier élément enfant de l' [table](#) élément.

Si une étape de cet algorithme nécessite que l' *élément actuel* soit **avancé jusqu'au prochain enfant du [table](#)** lorsqu'il n'y a pas d'enfant suivant, alors l'agent utilisateur doit sauter à l'étape étiquetée *end* , près de la fin de cet algorithme.

8. Tant que l' *élément courant* n'est pas l'un des éléments suivants, [faire avancer](#) l' *élément courant* jusqu'à l'enfant suivant de [table](#):
  - [colgroup](#)
  - [thead](#)
  - [tbody](#)
  - [tfoot](#)
  - [tr](#)

9. Si l' *élément actuel* est un colgroup, suivez ces sous-étapes :

- o *Groupes de colonnes* : Traite l' *élément courant* selon le cas approprié ci-dessous :

**Si l' *élément actuel* a des coléléments enfants**

Suivez ces étapes:

1. Soit  $x_{start}$  la valeur de  $x_{width}$  .
2. Laissez la *colonne actuelle* être le premier colélément enfant de l' colgroupélément.
3. *Columns* : si l' *élément de colonne actuel* col a un spanattribut, analysez sa valeur en utilisant les [règles d'analyse des entiers non négatifs](#) .

Si le résultat de l'analyse de la valeur n'est pas une erreur ou zéro, laissez *span* être cette valeur.

Sinon, si l' colélément n'a pas spand'attribut, ou si la tentative d'analyse de la valeur de l'attribut a entraîné une erreur ou zéro, alors laissez *span* être 1.

Si *span* est supérieur à 1000, laissez-le être 1000 à la place.

4. Augmenter *la largeur*  $x$  de *span* .
5. Laissez les dernières colonnes d'étendue du *tableau* correspondre à l' *élément de colonne actuel* . col
6. Si la *colonne actuelle* n'est pas le dernier colélément enfant de l' colgroupélément, laissez la *colonne actuelle* être le prochain colélément enfant de l' colgroupélément et revenez à l'étape intitulée *columns* .
7. Soit toutes les dernières colonnes du *tableau* de  $x = x_{start}$  à  $x = x_{width} - 1$  forment un nouveau groupe de colonnes , ancré à l'emplacement ( $x_{start}$  , 0), avec width  $x_{width} - x_{start}$  , correspondant à l' colgroupélément.

**Si l' *élément actuel* n'a pas cold'éléments enfants**

8. Si l' colgroupélément a un span attribut, analysez sa valeur en utilisant les [règles d'analyse des entiers non négatifs](#) .

Si le résultat de l'analyse de la valeur n'est pas une erreur ou zéro, laissez *span* être cette valeur.

Sinon, si l' colgroupélément n'a pas spand'attribut, ou si la tentative d'analyse de la valeur de l'attribut a entraîné une erreur ou zéro, alors laissez *span* être 1.

Si *span* est supérieur à 1000, laissez-le être 1000 à la place.

9. Augmenter la largeur *x* de *span* .

10. Laissez les dernières colonnes *span* du *tableau* former un nouveau groupe de colonnes , ancré à l'emplacement (  $x_{width} - span$  , 0), avec *width span* , correspondant à l' élément colgroup

- Avance l' *élément actuel* jusqu'à l'enfant suivant du table.
- Tant que l' *élément courant* n'est pas l'un des éléments suivants, faire avancer l' *élément courant* jusqu'à l'enfant suivant de table:

1. colgroup
2. thead
3. tbody
4. tfoot
5. tr

- Si l' *élément actuel* est un colgroupélément, passez à l'étape intitulée *groupes de colonnes* ci-dessus.

10. Soit  $y_{courant}$  nul.

11. Soit la *liste des cellules croissantes vers le bas* soit une liste vide.

12. *Lignes* : tant que l' *élément actuel* n'est pas l'un des éléments suivants, faire avancer l' *élément actuel* jusqu'à l'enfant suivant de table :

- thead
- tbody
- tfoot
- tr

13. Si l' *élément courant* est un tr, alors exécutez l' algorithme de traitement des lignes , avancez l' *élément courant* jusqu'au prochain enfant du table et revenez à l'étape intitulée *rows* .

14. Exécutez l' algorithme pour terminer un groupe de lignes .

15. Si l' *élément actuel* est un tfoot, ajoutez cet élément à la liste des *éléments en attente* tfoot , avancez l' *élément actuel* jusqu'au prochain enfant de table et revenez à l'étape intitulée *rows* .

16. L' *élément courant* est soit a thead soit a tbody.

Exécutez l' algorithme de traitement des groupes de lignes .

17. [Avance](#) l' *élément actuel* jusqu'à l'enfant suivant du [table](#).
18. Revenez à l'étape intitulée *rows*.
19. *Fin* : Pour chaque [tfoot](#) élément de la liste des *éléments en attente* [tfoot](#), dans [l'ordre de l'arborescence](#), exécutez l' [algorithme de traitement des groupes de lignes](#).
20. S'il existe une [ligne](#) ou [une colonne](#) dans *le tableau* contenant uniquement [des emplacements](#) auxquels aucune [cellule](#) n'est ancrée, il s'agit d'une [erreur de modèle de tableau](#).
21. Retournez *le tableau*.

L' **algorithme de traitement des groupes de lignes**, qui est appelé par l'ensemble des étapes ci-dessus pour le traitement des éléments [thead](#), [tbody](#) et [tfoot](#), est :

1. Soit  $y_{start}$  la valeur de  $y_{height}$ .
2. Pour chaque [tr](#) élément enfant de l'élément en cours de traitement, dans l'ordre de l'arborescence, exécutez l' [algorithme de traitement des lignes](#).
3. Si  $y_{height} > y_{start}$ , alors laissez toutes les dernières [lignes](#) du *tableau* de  $y = y_{start}$  à  $y = y_{height} - 1$  former un nouveau [groupe de lignes](#), ancré à l'emplacement avec la coordonnée  $(0, y_{start})$ , avec la hauteur  $y_{height} - y_{start}$ , correspondant à l'élément en cours de traitement.
4. Exécutez l' [algorithme pour terminer un groupe de lignes](#).

L' **algorithme de fin d'un groupe de lignes**, qui est appelé par l'ensemble des étapes ci-dessus lors du démarrage et de la fin d'un bloc de lignes, est :

1. Tant que  $y_{courant}$  est inférieur à  $y_{hauteur}$ , procédez comme suit :
  1. Exécutez l' [algorithme de croissance des cellules à croissance descendante](#).
  2. Augmentez *le*  $y_{courant}$  de 1.
2. Vide la *liste des cellules croissant vers le bas*.

L' **algorithme de traitement des lignes**, qui est appelé par l'ensemble des étapes ci-dessus pour le traitement [tr](#) des éléments, est :

1. Si  $y_{hauteur}$  est égale à  $y_{courant}$ , alors augmentez  $y_{hauteur}$  de 1. ( $y_{courant}$  n'est jamais *supérieur* à  $y_{hauteur}$ .)
2. Soit  $x_{courant}$  égal à 0.
3. Exécutez l' [algorithme de croissance des cellules à croissance descendante](#).



4. Si l' trélément en cours de traitement n'a pas d'élément enfant td ou th , augmentez  $y_{current}$  de 1, abandonnez cet ensemble d'étapes et revenez à l'algorithme ci-dessus.
5. Soit *la cellule actuelle* soit le premier td élément th enfant de l' trélément en cours de traitement.
6. *Cellules* : Alors que  $x_{courant}$  est inférieur à  $x_{largeur}$  et que l'emplacement avec les coordonnées (  $x_{courant}$  ,  $y_{courant}$  ) a déjà une cellule qui lui est assignée, augmentez  $x_{courant}$  de 1.
7. Si  $x_{courant}$  est égal à  $x_{largeur}$  , augmentez  $x_{largeur}$  de 1. (  $x_{courant}$  n'est jamais supérieur à  $x_{largeur}$  .)
8. Si la *cellule actuelle* a un colspan attribut, [analysez la valeur de cet attribut](#) et laissez *colspan* être le résultat.

Si l'analyse de cette valeur a échoué, ou a renvoyé zéro, ou si l'attribut est absent, alors laissez *colspan* être 1, à la place.

Si *colspan* est supérieur à 1000, laissez-le être 1000 à la place.

9. Si la *cellule actuelle* a un rowspan attribut, [analysez la valeur de cet attribut](#) et laissez *rowspan* être le résultat.

Si l'analyse de cette valeur a échoué ou si l'attribut est absent, laissez *rowspan* être 1 à la place.

Si *rowspan* est supérieur à 65534, laissez-le être 65534 à la place.

10. Si *rowspan* est égal à zéro et que le [document de nœudtable](#) de l'élément n'est pas défini sur [quirks mode](#) , alors laissez *cell grows down* être vrai et définissez *rowspan* sur 1. Sinon, laissez *cell grows down* être faux.
11. Si  $x_{width} < x_{current} + colspan$  , alors soit  $x_{width}$  soit  $x_{current} + colspan$  .
12. Si  $y_{hauteur} < y_{courant} + rowspan$  , alors laissez  $y_{hauteur}$  être  $y_{courant} + rowspan$  .

13. Supposons que les emplacements de coordonnées (  $x$  ,  $y$  ) tels que  $x_{courant} \leq x < x_{courant} + colspan$  et  $y_{courant} \leq y < y_{courant} + rowspan$  soient couverts par une nouvelle cellule *c* , ancrée à (  $x_{courant}$  ,  $y_{courant}$  ), qui a *width colspan* et *height rowspan* , correspondant à l'élément de *cellule actuel* .

Si l'élément de *cellule courant* th est un élément, soit cette nouvelle cellule *c* une cellule d'en-tête ; sinon, que ce soit une cellule de données.

Pour établir quelles cellules d'en-tête s'appliquent à l'élément de *cellule actuel* , utilisez l' [algorithme d'attribution des cellules d'en-tête](#) décrit dans la section suivante.

Si l'un des emplacements concernés était déjà couvert par une [cellule](#) , il s'agit d'une [erreur de modèle de table](#) . Ces emplacements ont maintenant deux cellules qui se chevauchent.

14. Si *la cellule croît vers le bas* est vraie, alors ajoutez le tuple  $\{ c, x_{\text{courant}}, \text{colspan} \}$  à la *liste des cellules qui croissent vers le bas* .
15. Augmenter  $x_{\text{courant}}$  par *colspan* .
16. Si *la cellule actuelle* est le dernier [td](#) ou [th](#) l'élément enfant de l' [tr](#) élément en cours de traitement, alors augmentez  $y_{\text{courant}}$  de 1, abandonnez cet ensemble d'étapes et revenez à l'algorithme ci-dessus.
17. Laissez *la cellule actuelle* être l'enfant suivant [td](#) ou [th](#) élément dans l' [tr](#) élément en cours de traitement.
18. Revenez à l'étape *cellules étiquetées* .

Lorsque les algorithmes ci-dessus nécessitent que l'agent utilisateur exécute l' **algorithme de croissance des cellules à croissance descendante** , l'agent utilisateur doit, pour chaque tuple  $\{ cell, cell_x, width \}$  dans la *liste des cellules à croissance descendante* , le cas échéant, étendre la [cellule](#) cellule de sorte qu'elle couvre également les emplacements de coordonnées  $(x, y_{\text{courant}})$  , où  $cellule_x \leq x < cellule_x + largeur$  .

#### 4.9.12.2 Formation de relations entre les cellules de données et les cellules d'en-tête

Chaque cellule peut se voir attribuer zéro ou plusieurs cellules d'en-tête. L' **algorithme d'attribution de cellules d'en-tête** à une *cellule principale* de cellule est le suivant.

1. Soit *la liste d'en-tête* une liste vide de cellules.
2. Soit  $(principal_x, principal_y)$  la coordonnée de la fente à laquelle la *cellule principale* est ancrée.
3. Si la *cellule principale* a un [headers](#) attribut spécifié
  1. Prenez la valeur de l' attribut *principal cell* et [headers](#) divisez -le en [ASCII whitespace](#) , laissant *id list* être la liste des jetons obtenus.
  2. Pour chaque jeton de la *liste d'identifiants* , si le premier élément avec [Document](#) un [ID](#) égal au jeton est une cellule de la même [table](#) et que cette cellule n'est pas la *cellule principale* , ajoutez cette cellule à la *liste d'en-tête* .

Si la *cellule principale* n'a pas d' [headers](#) attribut spécifié

3. Soit  $largeur_{principale}$  la largeur de la *cellule principale* .
  4. Soit  $hauteur_{principale}$  la hauteur de la *cellule principale* .
  5. Pour chaque valeur de  $y$  de  $principal_y$  à  $principal_y + hauteur_{principale} - 1$ , exécutez l' [algorithme interne pour scanner et affecter les cellules d'en-tête](#) , avec la *cellule principale* , la *liste d'en-tête* , la coordonnée initiale (  $principal_x$  ,  $y$  ) et les incréments  $\Delta x = -1$  et  $\Delta y = 0$  .
  6. Pour chaque valeur de  $x$  de  $x_{principale}$  à  $x_{principale} + largeur_{principale} - 1$ , exécutez l' [algorithme interne de numérisation et d'attribution des cellules d'en-tête](#) , avec la *cellule principale* , la *liste d'en-tête* , la coordonnée initiale (  $x$  ,  $principal_y$  ) et les incréments  $\Delta x = 0$  et  $\Delta y = -1$  .
  7. Si la *cellule principale* est ancrée dans un [groupe de lignes](#) , ajoutez toutes les cellules d'en-tête qui sont des [en-têtes de groupe de lignes](#) et sont ancrées dans le même groupe de lignes avec une coordonnée  $x$  inférieure ou égale à  $x_{principale} + largeur_{principale} - 1$  et un  $y$  - coordonnée inférieure ou égale à  $la_{principale}_y + hauteur_{principale} - 1$  à la *liste d'en-tête* .
  8. Si la *cellule principale* est ancrée dans un [groupe de colonnes](#) , ajoutez toutes les cellules d'en-tête qui sont [des en-têtes de groupe de colonnes](#) et sont ancrées dans le même groupe de colonnes avec une coordonnée  $x$  inférieure ou égale à  $x_{principale} + largeur_{principale} - 1$  et un  $y$  - coordonnée inférieure ou égale à  $la_{principale}_y + hauteur_{principale} - 1$  à la *liste d'en-tête* .
4. Supprimez toutes les [cellules vides](#) de la *liste d'en-tête* .
  5. Supprimez tous les doublons de la *liste d'en-tête* .
  6. Supprimez la *cellule principale* de la *liste d'en-tête* si elle s'y trouve.
  7. Attribuez les en-têtes de la *liste d'en-têtes* à la *cellule principale* .

L' **algorithme interne de balayage et d'attribution des cellules d'en-tête** , étant donné une *cellule principale* , une *liste d'en-tête* , une coordonnée initiale (  $x_{initial}$  ,  $y_{initial}$  ) et des incréments  $\Delta x$  et  $\Delta y$  , est le suivant :

1. Soit  $x$  égal à  $x_{initial}$  .
2. Soit  $y$  égal à  $y_{initial}$  .
3. Laissez les *en-têtes opaques* être une liste vide de cellules.
4. **Si la cellule principale est une cellule d'en-tête**  
 Soit dans le *bloc d'en-tête* la valeur true, et soit les *en-têtes du bloc d'en-tête actuel* soient une liste de cellules contenant uniquement la *cellule principale* .

**Sinon**

Laissez *in header block* être faux et laissez *les en-têtes du bloc d'en-tête actuel* être une liste vide de cellules.

5. *Boucle* : Incrémenter  $x$  de  $\Delta x$  ; incrémenter  $y$  de  $\Delta y$  .

Pour chaque invocation de cet algorithme, l'un de  $\Delta x$  et  $\Delta y$  sera -1, et l'autre sera 0.

6. Si  $x$  ou  $y$  sont inférieurs à 0, abandonnez cet algorithme interne.
7. S'il n'y a pas de fente de couverture de cellule (  $x$  ,  $y$  ), ou s'il y a plus d'une fente de couverture de cellule (  $x$  ,  $y$  ), retournez à la sous-étape étiquetée *boucle* .
8. Soit *la cellule actuelle* la cellule couvrant l'emplacement (  $x$  ,  $y$  ).

**9. Si *la cellule actuelle* est une cellule d'en-tête**

1. Défini *dans le bloc d'en-tête* sur vrai.
2. Ajouter *la cellule actuelle* aux *en-têtes du bloc d'en-tête actuel* .
3. Soit *bloqué* être faux.

**4. Si  $\Delta x$  vaut 0**

S'il y a des cellules dans la liste *des en-têtes opaques* ancrées avec la même coordonnée  $x$  que la *cellule actuelle* et avec la même largeur que *la cellule actuelle* , alors laissez *blocked* être vrai.

Si la *cellule actuelle* n'est pas un [en-tête de colonne](#) , laissez *blocked* être vrai.

**Si  $\Delta y$  vaut 0**

S'il y a des cellules dans la liste *des en-têtes opaques* ancrées avec la même coordonnée  $y$  que la *cellule actuelle* et avec la même hauteur que *la cellule actuelle* , alors laissez *blocked* être vrai.

Si la *cellule actuelle* n'est pas un [en-tête de ligne](#) , laissez *blocked* être vrai.

5. Si *bloqué* est faux, alors ajoutez la *cellule actuelle* à la *liste des en-têtes* .

**Si *la cellule actuelle* est une cellule de données et que le bloc d'en-tête est vrai**

Défini *dans le bloc d'en-tête* sur false. Ajoutez toutes les cellules des *en-têtes du bloc d'en-tête actuel* à la liste *des en-têtes opaques* et videz les *en-têtes de la liste de bloc d'en-tête actuelle* .

10. Revenez à l'étape intitulée *loop* .

Une cellule d'en-tête ancrée à l'emplacement avec la coordonnée (  $x, y$  ) avec largeur *largeur* et hauteur *hauteur* est dite être un **en-tête de colonne** si l'une des conditions suivantes est vraie :

- L' scope attribut de la cellule est à l' état de colonne , ou
- L' scope attribut de la cellule est dans l' état automatique et il n'y a aucune cellule de données dans aucune des cellules couvrant les emplacements avec les coordonnées  $y \dots y + hauteur - 1$  .

Une cellule d'en-tête ancrée à l'emplacement avec la coordonnée (  $x, y$  ) avec largeur *largeur* et hauteur *hauteur* est dite être un **en-tête de ligne** si l'une des conditions suivantes est vraie :

- L' scope attribut de la cellule est à l' état de ligne , ou
- L'attribut de la cellule scope est dans l' état automatique , la cellule n'est pas un en-tête de colonne et il n'y a aucune cellule de données dans aucune des cellules couvrant les emplacements avec les coordonnées  $x \dots x + largeur - 1$  .

Une cellule d'en-tête est dite être un **en-tête de groupe de colonnes** si son scope attribut est à l' état de groupe de colonnes .

Une cellule d'en-tête est dite être un **en-tête de groupe de lignes** si son scope attribut est à l' état de groupe de lignes .

Une cellule est dite **vide** si elle ne contient aucun élément et que son contenu textuel enfant , s'il y en a un, se compose uniquement d' espaces ASCII .

### 4.9.13 Exemples

*Cette section est non normative.*

Ce qui suit montre comment on peut baliser la partie inférieure du tableau 45 des *tables physiques du Smithsonian, Volume 71* :

```
<table>
  <caption>Specification values: <b>Steel</b>, <b>Castings</b>,
  Ann. A.S.T.M. A27-16, Class B;* P max. 0.06; S max.
  0.05.</caption>
  <thead>
    <tr>
      <th rowspan=2>Grade.</th>
      <th rowspan=2>Yield Point.</th>
      <th colspan=2>Ultimate tensile strength</th>
```

```

<th rowspan=2>Per cent elong. 50.8&nbsp;mm or 2&nbsp;in.</th>
<th rowspan=2>Per cent reduct. area.</th>
</tr>
<tr>
<th>kg/mm<sup>2</sup></th>
<th>lb/in<sup>2</sup></th>
</tr>
</thead>
<tbody>
<tr>
<td>Hard</td>
<td>0.45 ultimate</td>
<td>56.2</td>
<td>80,000</td>
<td>15</td>
<td>20</td>
</tr>
<tr>
<td>Medium</td>
<td>0.45 ultimate</td>
<td>49.2</td>
<td>70,000</td>
<td>18</td>
<td>25</td>
</tr>
<tr>
<td>Soft</td>
<td>0.45 ultimate</td>
<td>42.2</td>
<td>60,000</td>
<td>22</td>
<td>30</td>
</tr>
</tbody>
</table>

```

Ce tableau pourrait ressembler à ceci :

Valeurs de spécification : **Acier , Pièces moulées** , Ann. ASTM A27-16, Classe B;\*  
P max. 0,06 ; S max. 0,05.

Grade.	Seuil de rentabilité.	Résistance à la traction ultime		Pour cent allongé. 50, 8 mm ou 2 pouces	% de réduction. zone.
		kg/ mm <sup>2</sup>	lb/ po <sup>2</sup>		
Dur	0,45 ultime	56.2	80000	15	20
Moyen	0,45 ultime	49.2	70000	18	25
Doux	0,45 ultime	42.2	60000	22	30

Ce qui suit montre comment on peut baliser le tableau de la marge brute à la page 46 du dossier 10-K d'Apple, Inc pour l'exercice 2008 :

```

<table>
  <thead>
    <tr>
      <th>
        <th>2008
        <th>2007
        <th>2006
      <tbody>
        <tr>
          <th>Net sales
          <td>$ 32,479
          <td>$ 24,006
          <td>$ 19,315
        <tr>
          <th>Cost of sales
          <td> 21,334
          <td> 15,852
          <td> 13,717

```

```

<tbody>
  <tr>
    <th>Gross margin
    <td>$ 11,145
    <td>$ 8,154
    <td>$ 5,598
  </tr>
</tbody>
<tfoot>
  <tr>
    <th>Gross margin percentage
    <td>34.3%
    <td>34.0%
    <td>29.0%
  </tr>
</tfoot>
</table>

```

Ce tableau pourrait ressembler à ceci :

	2008	2007	2006
<b>Ventes nettes</b>	32 479 \$	24 006 \$	19 315 \$
<b>Coût des ventes</b>	21 334	15 852	13 717
<b>marge brute</b>	11 145 \$	8 154 \$	5 598 \$
<b>Pourcentage de marge brute</b>	34,3 %	34,0 %	29,0 %

Ce qui suit montre comment on peut baliser le tableau des dépenses d'exploitation à partir du bas sur la même page de ce document :

```

<table>
  <colgroup> <col>
  <colgroup> <col> <col> <col>
  <thead>
    <tr> <th> <th>2008 <th>2007 <th>2006
  </thead>
  <tbody>
    <tr> <th scope=rowgroup> Research and development
      <td> $ 1,109 <td> $ 782 <td> $ 712
    <tr> <th scope=row> Percentage of net sales
      <td> 3.4% <td> 3.3% <td> 3.7%
    <tbody>
      <tr> <th scope=rowgroup> Selling, general, and administrative
        <td> $ 3,761 <td> $ 2,963 <td> $ 2,433

```



```

<tr> <th scope=row> Percentage of net sales
    <td> 11.6% <td> 12.3% <td> 12.6%
</table>

```

Ce tableau pourrait ressembler à ceci :

	2008	2007	2006
<b>Recherche et développement</b>	1 109 \$	782 \$	712 \$
<b>Pourcentage des ventes nettes</b>	3,4 %	3,3 %	3,7 %
<b>Frais de vente, généraux et administratifs</b>	3 761 \$	2 963 \$	2 433 \$
<b>Pourcentage des ventes nettes</b>	11,6 %	12,3 %	12,6 %

## 4.10 Formulaires



### 4.10.1 Présentation

*Cette section est non normative.*

Un formulaire est un composant d'une page Web qui comporte des contrôles de formulaire, tels que du texte, des boutons, des cases à cocher, une plage ou des contrôles de sélecteur de couleurs. Un utilisateur peut interagir avec un tel formulaire, fournissant des données qui peuvent ensuite être envoyées au serveur pour un traitement ultérieur (par exemple, retour des résultats d'une recherche ou d'un calcul). Aucun script côté client n'est nécessaire dans de nombreux cas, bien qu'une API soit disponible afin que les scripts puissent améliorer l'expérience utilisateur ou utiliser des formulaires à des fins autres que la soumission de données à un serveur.

L'écriture d'un formulaire se compose de plusieurs étapes, qui peuvent être effectuées dans n'importe quel ordre : écriture de l'interface utilisateur, implémentation du traitement côté serveur et configuration de l'interface utilisateur pour communiquer avec le serveur.

#### 4.10.1.1 Ecrire l'interface utilisateur d'un formulaire

*Cette section est non normative.*

Pour les besoins de cette brève introduction, nous allons créer un formulaire de commande de pizza.

Tout formulaire commence par un `<form>` élément, à l'intérieur duquel sont placés les contrôles. La plupart des contrôles sont représentés par l' `<input>` élément, qui fournit par défaut un contrôle de texte. Pour étiqueter un contrôle, l' `<label>` élément est utilisé ; le texte de l'étiquette et le contrôle lui-même vont à l'intérieur de l' `<label>` élément. Chaque partie d'un formulaire est considérée comme un `<paragraphe>` et est généralement séparée des autres parties à l'aide `<p>` d'éléments. En mettant cela ensemble, voici comment on pourrait demander le nom du client :

```
<form>
  <p><label>Customer name: <input></label></p>
</form>
```

Pour permettre à l'utilisateur de sélectionner la taille de la pizza, nous pouvons utiliser un ensemble de boutons radio. Les boutons radio utilisent également l' `<input>` élément, cette fois avec un `type` attribut avec la valeur `radio`. Pour que les boutons radio fonctionnent en groupe, un nom commun leur est attribué à l'aide de l' `name` attribut. Pour regrouper un lot de champs, comme ici les boutons radio, on peut utiliser l' `<fieldset>` élément . Le titre d'un tel groupe de contrôles est donné par le premier élément du `<fieldset>`, qui doit être un `<legend>` élément.

```
<form>
  <p><label>Customer name: <input></label></p>
  <fieldset>
    <legend> Pizza Size </legend>
    <p><label> <input type=radio name=size> Small </label></p>
    <p><label> <input type=radio name=size> Medium </label></p>
    <p><label> <input type=radio name=size> Large </label></p>
  </fieldset>
</form>
```

*Les modifications par rapport à l'étape précédente sont mises en surbrillance.*

Pour choisir les garnitures, nous pouvons utiliser des cases à cocher. Ceux-ci utilisent l' `<input>` élément avec un `type` attribut avec la valeur `checkbox`:

```
<form>
  <p><label>Customer name: <input></label></p>
  <fieldset>
    <legend> Pizza Size </legend>
    <p><label> <input type=radio name=size> Small </label></p>
    <p><label> <input type=radio name=size> Medium </label></p>
    <p><label> <input type=radio name=size> Large </label></p>
```

```

</fieldset>
<fieldset>
  <legend> Pizza Toppings </legend>
  <p><label> <input type=checkbox> Bacon </label></p>
  <p><label> <input type=checkbox> Extra Cheese </label></p>
  <p><label> <input type=checkbox> Onion </label></p>
  <p><label> <input type=checkbox> Mushroom </label></p>
</fieldset>
</form>

```

La pizzeria pour laquelle ce formulaire est rédigé fait toujours des erreurs, elle a donc besoin d'un moyen de contacter le client. À cette fin, nous pouvons utiliser des contrôles de formulaire spécifiquement pour les numéros de téléphone ( input éléments dont type l'attribut est défini sur tel) et les adresses e-mail ( input éléments dont l' type attribut est défini sur email) :

```

<form>
  <p><label>Customer name: <input></label></p>
  <p><label>Telephone: <input type=tel></label></p>
  <p><label>Email address: <input type=email></label></p>
  <fieldset>
    <legend> Pizza Size </legend>
    <p><label> <input type=radio name=size> Small </label></p>
    <p><label> <input type=radio name=size> Medium </label></p>
    <p><label> <input type=radio name=size> Large </label></p>
  </fieldset>
  <fieldset>
    <legend> Pizza Toppings </legend>
    <p><label> <input type=checkbox> Bacon </label></p>
    <p><label> <input type=checkbox> Extra Cheese </label></p>
    <p><label> <input type=checkbox> Onion </label></p>
    <p><label> <input type=checkbox> Mushroom </label></p>
  </fieldset>
</form>

```

Nous pouvons utiliser un input élément avec son type attribut défini sur time pour demander un délai de livraison. Beaucoup de ces contrôles de formulaire ont des attributs pour contrôler exactement quelles valeurs peuvent être spécifiées ; dans ce cas, trois attributs particulièrement intéressants sont min, max et step. Ceux-ci définissent le temps minimum, le temps maximum et l'intervalle entre les valeurs autorisées (en secondes). Cette pizzeria ne livre qu'entre 11h et 21h, et ne promet rien de mieux que des tranches de 15 minutes, que l'on peut baliser comme suit :

```

<form>

```

```

<p><label>Customer name: <input></label></p>
<p><label>Telephone: <input type=tel></label></p>
<p><label>Email address: <input type=email></label></p>
<fieldset>
  <legend> Pizza Size </legend>
  <p><label> <input type=radio name=size> Small </label></p>
  <p><label> <input type=radio name=size> Medium </label></p>
  <p><label> <input type=radio name=size> Large </label></p>
</fieldset>
<fieldset>
  <legend> Pizza Toppings </legend>
  <p><label> <input type=checkbox> Bacon </label></p>
  <p><label> <input type=checkbox> Extra Cheese </label></p>
  <p><label> <input type=checkbox> Onion </label></p>
  <p><label> <input type=checkbox> Mushroom </label></p>
</fieldset>
<p><label>Preferred delivery time: <input type=time min="11:00"
max="21:00" step="900"></label></p>
</form>

```

L' textarea élément peut être utilisé pour fournir un contrôle de texte multiligne. Dans ce cas, nous allons l'utiliser pour fournir un espace au client pour donner des instructions de livraison :

```

<form>
  <p><label>Customer name: <input></label></p>
  <p><label>Telephone: <input type=tel></label></p>
  <p><label>Email address: <input type=email></label></p>
  <fieldset>
    <legend> Pizza Size </legend>
    <p><label> <input type=radio name=size> Small </label></p>
    <p><label> <input type=radio name=size> Medium </label></p>
    <p><label> <input type=radio name=size> Large </label></p>
  </fieldset>
  <fieldset>
    <legend> Pizza Toppings </legend>
    <p><label> <input type=checkbox> Bacon </label></p>
    <p><label> <input type=checkbox> Extra Cheese </label></p>
    <p><label> <input type=checkbox> Onion </label></p>
    <p><label> <input type=checkbox> Mushroom </label></p>
  </fieldset>

```

```

<p><label>Preferred delivery time: <input type=time min="11:00"
max="21:00" step="900"></label></p>
<p><label>Delivery instructions: <textarea></textarea></label></p>
</form>

```

Enfin, pour rendre le formulaire soumis, nous utilisons l' button élément :

```

<form>
<p><label>Customer name: <input></label></p>
<p><label>Telephone: <input type=tel></label></p>
<p><label>Email address: <input type=email></label></p>
<fieldset>
<legend> Pizza Size </legend>
<p><label> <input type=radio name=size> Small </label></p>
<p><label> <input type=radio name=size> Medium </label></p>
<p><label> <input type=radio name=size> Large </label></p>
</fieldset>
<fieldset>
<legend> Pizza Toppings </legend>
<p><label> <input type=checkbox> Bacon </label></p>
<p><label> <input type=checkbox> Extra Cheese </label></p>
<p><label> <input type=checkbox> Onion </label></p>
<p><label> <input type=checkbox> Mushroom </label></p>
</fieldset>
<p><label>Preferred delivery time: <input type=time min="11:00"
max="21:00" step="900"></label></p>
<p><label>Delivery instructions: <textarea></textarea></label></p>
<p><button>Submit order</button></p>
</form>

```

#### 4.10.1.2 Implémentation du traitement côté serveur pour un formulaire

*Cette section est non normative.*

Les détails exacts pour écrire un processeur côté serveur sont hors de portée de cette spécification. Pour les besoins de cette introduction, nous supposons que le script à <https://pizza.example.com/order.cgi> est configuré pour accepter les soumissions utilisant le application/x-www-form-urlencoded format, en attendant les paramètres suivants envoyés dans un corps HTTP POST :

**custname**

Nom du client

`custtel`

Numéro de téléphone du client

`custemail`

Adresse e-mail du client

`size`

La taille de la pizza, soit `small`, `medium`, ou `large`

`topping`

Une garniture, spécifiée une fois pour chaque garniture sélectionnée, les valeurs autorisées étant `bacon`, `cheese`, `onion` et `mushroom`

`delivery`

Le délai de livraison demandé

`comments`

Les consignes de livraison

#### 4.10.1.3 Configurer un formulaire pour communiquer avec un serveur

*Cette section est non normative.*

Les soumissions de formulaires sont exposées aux serveurs de diverses manières, le plus souvent sous forme de requêtes HTTP GET ou POST. Pour spécifier la méthode exacte utilisée, l' `method` attribut est spécifié sur l' `form` élément. Cela ne précise cependant pas comment les données du formulaire sont encodées ; pour le spécifier, vous utilisez l' `enctype` attribut. Vous devez également spécifier l' `URL` du service qui gèrera les données soumises, à l'aide de l' `action` attribut.

Pour chaque contrôle de formulaire que vous souhaitez soumettre, vous devez ensuite donner un nom qui sera utilisé pour faire référence aux données dans la soumission. Nous avons déjà spécifié le nom du groupe de boutons radio ; le même attribut ( `name` ) spécifie également le nom de la soumission. Les boutons radio peuvent être distingués les uns des autres dans la soumission en leur donnant des valeurs différentes, à l'aide de l' `value` attribut.

Plusieurs contrôles peuvent avoir le même nom ; par exemple, ici, nous donnons le même nom à toutes les cases à cocher, et le serveur distingue quelle case a été cochée en voyant quelles valeurs sont soumises avec ce nom - comme les boutons radio, ils reçoivent également des valeurs uniques avec l'attribut `value`.

Compte tenu des paramètres de la section précédente, tout cela devient :

```
<form method="post"
      enctype="application/x-www-form-urlencoded"
      action="https://pizza.example.com/order.cgi">
```

```

<p><label>Customer name: <input name="custname"></label></p>
<p><label>Telephone: <input type=tel name="custtel"></label></p>
<p><label>Email address: <input type=email
name="custemail"></label></p>
<fieldset>
  <legend> Pizza Size </legend>
  <p><label> <input type=radio name=size value="small"> Small
</label></p>
  <p><label> <input type=radio name=size value="medium"> Medium
</label></p>
  <p><label> <input type=radio name=size value="large"> Large
</label></p>
</fieldset>
<fieldset>
  <legend> Pizza Toppings </legend>
  <p><label> <input type=checkbox name="topping" value="bacon">
Bacon </label></p>
  <p><label> <input type=checkbox name="topping" value="cheese">
Extra Cheese </label></p>
  <p><label> <input type=checkbox name="topping" value="onion">
Onion </label></p>
  <p><label> <input type=checkbox name="topping" value="mushroom">
Mushroom </label></p>
</fieldset>
<p><label>Preferred delivery time: <input type=time min="11:00"
max="21:00" step="900" name="delivery"></label></p>
<p><label>Delivery instructions: <textarea
name="comments"></textarea></label></p>
<p><button>Submit order</button></p>
</form>

```

*Il n'y a pas de signification particulière à la façon dont certains des attributs ont leurs valeurs citées et d'autres non. La syntaxe HTML permet une variété de manières également valables de spécifier des attributs, comme indiqué [dans la section syntaxe](#) .*

Par exemple, si le client a saisi "Denise Lawrence" comme nom, "555-321-8642" comme numéro de téléphone, n'a pas spécifié d'adresse e-mail, a demandé une pizza de taille moyenne, a sélectionné les garnitures Extra Fromage et Champignons, entré une heure de livraison de 19 heures et laissé le contrôle du texte des instructions de livraison vide, l'agent utilisateur soumettrait ce qui suit au service Web en ligne :

```

custname=Denise+Lawrence&custtel=555-321-
8642&custemail=&size=medium&topping=cheese&topping=mushroom&deliver
y=19%3A00&comments=

```

#### 4.10.1.4 Validation du formulaire côté client



*Cette section est non normative.*

Les formulaires peuvent être annotés de manière à ce que l'agent utilisateur vérifie l'entrée de l'utilisateur avant que le formulaire ne soit soumis. Le serveur doit toujours vérifier que l'entrée est valide (puisque les utilisateurs hostiles peuvent facilement contourner la validation du formulaire), mais cela permet à l'utilisateur d'éviter l'attente encourue en faisant en sorte que le serveur soit le seul vérificateur de l'entrée de l'utilisateur.

L'annotation la plus simple est l' required attribut, qui peut être spécifié sur input des éléments pour indiquer que le formulaire ne doit pas être soumis tant qu'une valeur n'est pas donnée. En ajoutant cet attribut aux champs du nom du client, de la taille de la pizza et du délai de livraison, nous permettons à l'agent utilisateur d'informer l'utilisateur lorsqu'il soumet le formulaire sans remplir ces champs :

```
<form method="post"
      enctype="application/x-www-form-urlencoded"
      action="https://pizza.example.com/order.cgi">
  <p><label>Customer name: <input name="custname"
required></label></p>
  <p><label>Telephone: <input type=tel name="custtel"></label></p>
  <p><label>Email address: <input type=email
name="custemail"></label></p>
  <fieldset>
    <legend> Pizza Size </legend>
    <p><label> <input type=radio name=size required value="small">
Small </label></p>
    <p><label> <input type=radio name=size required value="medium">
Medium </label></p>
    <p><label> <input type=radio name=size required value="large">
Large </label></p>
  </fieldset>
  <fieldset>
    <legend> Pizza Toppings </legend>
    <p><label> <input type=checkbox name="topping" value="bacon">
Bacon </label></p>
    <p><label> <input type=checkbox name="topping" value="cheese">
Extra Cheese </label></p>
    <p><label> <input type=checkbox name="topping" value="onion">
Onion </label></p>
    <p><label> <input type=checkbox name="topping" value="mushroom">
Mushroom </label></p>
```



```

</fieldset>

<p><label>Preferred delivery time: <input type=time min="11:00"
max="21:00" step="900" name="delivery" required></label></p>

<p><label>Delivery instructions: <textarea
name="comments"></textarea></label></p>

<p><button>Submit order</button></p>

</form>

```

Il est également possible de limiter la longueur de l'entrée, en utilisant l' maxlength attribut. En ajoutant ceci à l' textarea élément, nous pouvons limiter les utilisateurs à 1000 caractères, les empêchant d'écrire d'énormes essais aux chauffeurs-livreurs occupés au lieu de rester concentrés et directs :

```

<form method="post"
      enctype="application/x-www-form-urlencoded"
      action="https://pizza.example.com/order.cgi">
  <p><label>Customer name: <input name="custname"
required></label></p>
  <p><label>Telephone: <input type=tel name="custtel"></label></p>
  <p><label>Email address: <input type=email
name="custemail"></label></p>
  <fieldset>
    <legend> Pizza Size </legend>
    <p><label> <input type=radio name=size required value="small">
Small </label></p>
    <p><label> <input type=radio name=size required value="medium">
Medium </label></p>
    <p><label> <input type=radio name=size required value="large">
Large </label></p>
  </fieldset>
  <fieldset>
    <legend> Pizza Toppings </legend>
    <p><label> <input type=checkbox name="topping" value="bacon">
Bacon </label></p>
    <p><label> <input type=checkbox name="topping" value="cheese">
Extra Cheese </label></p>
    <p><label> <input type=checkbox name="topping" value="onion">
Onion </label></p>
    <p><label> <input type=checkbox name="topping" value="mushroom">
Mushroom </label></p>
  </fieldset>
  <p><label>Preferred delivery time: <input type=time min="11:00"
max="21:00" step="900" name="delivery" required></label></p>
  <p><label>Delivery instructions: <textarea name="comments"
maxlength=1000></textarea></label></p>
  <p><button>Submit order</button></p>

```

```
</form>
```

Lorsqu'un formulaire est soumis, invalid des événements sont déclenchés à chaque contrôle de formulaire non valide. Cela peut être utile pour afficher un résumé des problèmes avec le formulaire, car généralement le navigateur lui-même ne signalera qu'un seul problème à la fois.

#### 4.10.1.5 Activation du remplissage automatique côté client des contrôles de formulaire

*Cette section est non normative.*

Certains navigateurs tentent d'aider l'utilisateur en remplissant automatiquement les contrôles de formulaire plutôt que de demander à l'utilisateur de ressaisir ses informations à chaque fois. Par exemple, un champ demandant le numéro de téléphone de l'utilisateur peut être rempli automatiquement avec le numéro de téléphone de l'utilisateur.

Pour aider l'agent utilisateur avec cela, l' autocomplete attribut peut être utilisé pour décrire le but du champ. Dans le cas de ce formulaire, nous avons trois champs qui peuvent être utilement annotés de cette manière : les informations sur le destinataire de la pizza. L'ajout de ces informations ressemble à ceci :

```
<form method="post"
  enctype="application/x-www-form-urlencoded"
  action="https://pizza.example.com/order.cgi">
  <p><label>Customer name: <input name="custname" required
    autocomplete="shipping name"></label></p>
  <p><label>Telephone: <input type="tel" name="custtel"
    autocomplete="shipping tel"></label></p>
  <p><label>Email address: <input type="email" name="custemail"
    autocomplete="shipping email"></label></p>
  <fieldset>
    <legend> Pizza Size </legend>
    <p><label> <input type="radio" name="size" required value="small">
    Small </label></p>
    <p><label> <input type="radio" name="size" required value="medium">
    Medium </label></p>
    <p><label> <input type="radio" name="size" required value="large">
    Large </label></p>
  </fieldset>
  <fieldset>
    <legend> Pizza Toppings </legend>
    <p><label> <input type="checkbox" name="topping" value="bacon">
    Bacon </label></p>
```

```

    <p><label> <input type=checkbox name="topping" value="cheese">
Extra Cheese </label></p>

    <p><label> <input type=checkbox name="topping" value="onion">
Onion </label></p>

    <p><label> <input type=checkbox name="topping" value="mushroom">
Mushroom </label></p>

</fieldset>

    <p><label>Preferred delivery time: <input type=time min="11:00"
max="21:00" step="900" name="delivery" required></label></p>

    <p><label>Delivery instructions: <textarea name="comments"
maxlength=1000></textarea></label></p>

    <p><button>Submit order</button></p>

</form>

```

#### 4.10.1.6 Améliorer l'expérience utilisateur sur les appareils mobiles

*Cette section est non normative.*

Certains appareils, notamment ceux dotés de claviers virtuels, peuvent offrir à l'utilisateur de multiples modalités de saisie. Par exemple, lors de la saisie d'un numéro de carte de crédit, l'utilisateur peut souhaiter ne voir que les touches des chiffres 0 à 9, tandis que lors de la saisie de son nom, il peut souhaiter voir un champ de formulaire qui, par défaut, met chaque mot en majuscule.

À l'aide de l' [inputmode](#) attribut, nous pouvons sélectionner les modalités d'entrée appropriées :

```

<form method="post"
    enctype="application/x-www-form-urlencoded"
    action="https://pizza.example.com/order.cgi">
    <p><label>Customer name: <input name="custname" required
autocomplete="shipping name"></label></p>

    <p><label>Telephone: <input type=tel name="custtel"
autocomplete="shipping tel"></label></p>

    <p><label>Buzzer code: <input name="custbuzz"


```

```

    <p><label> <input type=radio name=size required value="large">
    Large </label></p>
  </fieldset>

  <fieldset>

    <legend> Pizza Toppings </legend>

    <p><label> <input type=checkbox name="topping" value="bacon">
    Bacon </label></p>

    <p><label> <input type=checkbox name="topping" value="cheese">
    Extra Cheese </label></p>

    <p><label> <input type=checkbox name="topping" value="onion">
    Onion </label></p>

    <p><label> <input type=checkbox name="topping" value="mushroom">
    Mushroom </label></p>

  </fieldset>

  <p><label>Preferred delivery time: <input type=time min="11:00"
  max="21:00" step="900" name="delivery" required></label></p>

  <p><label>Delivery instructions: <textarea name="comments"
  maxlength=1000></textarea></label></p>

  <p><button>Submit order</button></p>
</form>

```

#### 4.10.1.7 La différence entre le type de champ, le nom du champ de remplissage automatique et la modalité de saisie

*Cette section est non normative.*

Les attributs type, autocomplete et inputmode peuvent sembler similaires à confusion. Par exemple, dans les trois cas, la chaîne "email" est une valeur valide. Cette section tente d'illustrer la différence entre les trois attributs et fournit des conseils suggérant comment les utiliser.

L' type attribut sur input les éléments décide du type de contrôle que l'agent utilisateur utilisera pour exposer le champ. Choisir entre différentes valeurs de cet attribut revient au même que choisir d'utiliser un input élément, un textarea élément, un select élément, etc.

L' autocomplete attribut, en revanche, décrit ce que représente réellement la valeur que l'utilisateur entrera. Choisir entre différentes valeurs de cet attribut revient au même que choisir le libellé de l'élément.

Considérons d'abord les numéros de téléphone. Si une page demande un numéro de téléphone à l'utilisateur, le bon contrôle de formulaire à utiliser est <input type=tel>. Cependant, la autocomplete valeur à utiliser dépend du numéro de téléphone demandé par la page, s'ils attendent un numéro de téléphone au format international ou simplement au format local, etc.

Par exemple, une page faisant partie d'un processus de paiement sur un site de commerce électronique pour un client achetant un cadeau à envoyer à un ami peut avoir besoin à la fois du numéro de téléphone de l'acheteur (en cas de problème de paiement) et du numéro de téléphone de l'ami ( en cas de problème de livraison). Si le site attend des numéros de téléphone internationaux (avec le préfixe du code du pays), cela pourrait donc ressembler à ceci :

```
<p><label>Your phone number: <input type=tel name=custtel  
autocomplete="billing tel"></label>  
  
<p><label>Recipient's phone number: <input type=tel name=shiptel  
autocomplete="shipping tel"></label>  
  
<p>Please enter complete phone numbers including the country code  
prefix, as in "+1 555 123 4567".
```

Mais si le site ne prend en charge que les clients et destinataires britanniques, il pourrait plutôt ressembler à ceci (notez l'utilisation de tel-national plutôt que tel):

```
<p><label>Your phone number: <input type=tel name=custtel  
autocomplete="billing tel-national"></label>  
  
<p><label>Recipient's phone number: <input type=tel name=shiptel  
autocomplete="shipping tel-national"></label>  
  
<p>Please enter complete UK phone numbers, as in "(01632) 960 123".
```

Maintenant, considérez les langues préférées d'une personne. La bonne autocomplete valeur est language. Cependant, il peut y avoir un certain nombre de contrôles de formulaire différents utilisés à cette fin : un contrôle de texte ( <input type=text> ), une liste déroulante ( <select> ), des boutons radio ( <input type=radio> ), etc. Cela dépend uniquement du type d'interface souhaité.

Enfin, pensez aux noms. Si une page ne demande qu'un seul nom à l'utilisateur, le contrôle pertinent est <input type=text>. Si la page demande le nom complet de l'utilisateur, la autocomplete valeur appropriée est name.

```
<p><label>Japanese name:<input name="j" type="text"  
autocomplete="section-jp name"></label>  
  
<label>Romanized name: <input name="e" type="text"  
autocomplete="section-en name"></label>
```

Dans cet exemple, les section-\* mots-clés " " dans les autocomplete valeurs des attributs indiquent à l'agent utilisateur que les deux champs attendent des noms *différents* . Sans eux, l'agent utilisateur pourrait remplir automatiquement le deuxième champ avec la valeur donnée dans le premier champ lorsque l'utilisateur a donné une valeur au premier champ.

*Les parties "-jp" et "-en" des mots clés sont opaques pour l'agent utilisateur ; l'agent utilisateur ne peut pas deviner, à partir de ceux-ci, que les deux noms sont censés être respectivement en japonais et en anglais.*

Indépendamment des choix concernant type et autocomplete, l' inputmode attribut décide du type de modalité de saisie (par exemple, clavier virtuel) à utiliser, lorsque le contrôle est un contrôle de texte.

Tenez compte des numéros de carte de crédit. Le type d'entrée approprié *n'est pas* `<input type=number>`, [comme expliqué ci-dessous](#) ; c'est plutôt `<input type=text>`. Pour encourager l'agent utilisateur à utiliser de toute façon une modalité de saisie numérique (par exemple, un clavier virtuel affichant uniquement des chiffres), la page utiliserait

```
<p><label>Credit card number:
      <input name="cc" type="text" inputmode="numeric"
pattern="[0-9]{8,19}" autocomplete="cc-number">
</label></p>
```

#### 4.10.1.8 Formats de date, d'heure et de nombre

*Cette section est non normative.*

Dans cet exemple de livraison de pizza, les heures sont spécifiées au format "HH:MM" : deux chiffres pour l'heure, au format 24 heures, et deux chiffres pour l'heure. (Les secondes peuvent également être spécifiées, bien qu'elles ne soient pas nécessaires dans cet exemple.)

Dans certains paramètres régionaux, cependant, les heures sont souvent exprimées différemment lorsqu'elles sont présentées aux utilisateurs. Par exemple, aux États-Unis, il est encore courant d'utiliser l'horloge de 12 heures avec un indicateur am/pm, comme dans "2pm". En France, il est courant de séparer les heures des minutes à l'aide d'un caractère "h", comme dans "14h00".

Des problèmes similaires existent avec les dates, avec la complication supplémentaire que même l'ordre des composants n'est pas toujours cohérent - par exemple, à Chypre, le premier février 2003 serait généralement écrit "1/2/03", alors que cette même date au Japon serait généralement écrit comme "2003年02月01日" - et même avec des nombres, où les paramètres régionaux diffèrent, par exemple, dans la ponctuation utilisée comme séparateur décimal et séparateur de milliers.

Il est donc important de distinguer les formats d'heure, de date et de nombre utilisés en HTML et dans les soumissions de formulaires, qui sont toujours les formats définis dans cette spécification (et basés sur la norme ISO 8601 bien établie pour les formats de date et d'heure lisibles par ordinateur ), à partir des formats d'heure, de date et de nombre présentés à l'utilisateur par le navigateur et acceptés comme entrée de l'utilisateur par le navigateur.

Le format utilisé "sur le fil", c'est-à-dire dans le balisage HTML et dans les soumissions de formulaires, est destiné à être lisible par ordinateur et cohérent quel

que soit le paramètre régional de l'utilisateur. Les dates, par exemple, sont toujours écrites au format "AAAA-MM-JJ", comme dans "2003-02-01". Alors que certains utilisateurs peuvent voir ce format, d'autres peuvent le voir comme "01.02.2003" ou "1er février 2003".

L'heure, la date ou le numéro donné par la page au format wire est ensuite traduit dans la présentation préférée de l'utilisateur (basée sur les préférences de l'utilisateur ou sur les paramètres régionaux de la page elle-même), avant d'être affiché à l'utilisateur. De même, après que l'utilisateur a entré une heure, une date ou un nombre en utilisant son format préféré, l'agent utilisateur le reconvertit au format filaire avant de le mettre dans le DOM ou de le soumettre.

Cela permet aux scripts dans les pages et sur les serveurs de traiter les heures, les dates et les nombres de manière cohérente sans avoir à prendre en charge des dizaines de formats différents, tout en répondant aux besoins des utilisateurs.

Voir aussi les [notes d'implémentation](#) concernant la localisation des contrôles de formulaire.

## 4.10.2 Catégories

Principalement pour des raisons historiques, les éléments de cette section appartiennent à plusieurs catégories qui se chevauchent (mais légèrement différentes) en plus des catégories habituelles telles que le [contenu de flux](#) , le [contenu de phrasé](#) et le [contenu interactif](#) .

Un certain nombre d'éléments sont **des éléments associés au formulaire** , ce qui signifie qu'ils peuvent avoir un [propriétaire de formulaire](#) .

- [button](#)
- [fieldset](#)
- [input](#)
- [object](#)
- [output](#)
- [select](#)
- [textarea](#)
- [img](#)
- [éléments personnalisés associés au formulaire](#)

Les [éléments associés au formulaire](#) se répartissent en plusieurs sous-catégories :

### ***Eléments listés***

Désigne les éléments répertoriés dans les API [form.elements](#) et [fieldset.elements](#). Ces éléments ont également un [form](#)attribut content et un [form](#)attribut IDL correspondant, qui permettent aux auteurs de spécifier un [propriétaire de formulaire](#) explicite .

- [button](#)
- [fieldset](#)
- [input](#)
- [object](#)
- [output](#)
- [select](#)
- [textarea](#)
- [éléments personnalisés associés au formulaire](#)

### **Éléments à soumettre**

Désigne les éléments qui peuvent être utilisés pour [construire la liste d'entrées](#) lorsqu'un [form](#)élément est [soumis](#) .

- [button](#)
- [input](#)
- [select](#)
- [textarea](#)
- [éléments personnalisés associés au formulaire](#)

Certains [éléments pouvant être soumis](#) peuvent être, selon leurs attributs, **des boutons** . La prose ci-dessous définit quand un élément est un bouton. Certains boutons sont spécifiquement **des boutons d'envoi** .

### **Éléments réinitialisables**

Désigne les éléments qui peuvent être affectés lorsqu'un [form](#)élément est [réinitialisé](#) .

- [input](#)
- [output](#)
- [select](#)
- [textarea](#)
- [éléments personnalisés associés au formulaire](#)

### **Éléments hérités de la capitalisation automatique**

Indique les éléments qui héritent de l' [autocapitalize](#) attribut de leur [propriétaire de formulaire](#) .

- [button](#)
- [fieldset](#)
- [input](#)
- [output](#)
- [select](#)
- [textarea](#)

Certains éléments, pas tous [associés au formulaire](#) , sont classés comme **éléments étiquetables** . Ce sont des éléments qui peuvent être associés à un [label](#)élément.

- [button](#)
- [input](#)(si l' [type](#)attribut n'est pas dans l' état [Masqué](#) )
- [meter](#)
- [output](#)



- [progress](#)
- [select](#)
- [textarea](#)
- [éléments personnalisés associés au formulaire](#)

#### 4.10.3 L' **form**élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de flux](#) est attendu.

##### Modèle de contenu :

[Contenu du flux](#) , mais sans [form](#) descendants d'éléments.

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
[accept-charset](#)— Encodages de caractères à utiliser pour [la soumission de formulaire](#)  
[action](#)— [URL](#) à utiliser pour [la soumission du formulaire](#)  
[autocomplete](#)— Paramètre par défaut de la fonction de remplissage automatique pour les contrôles du formulaire  
[enctype](#)— Type d'encodage [de la liste d'entrées](#) à utiliser pour [la soumission du formulaire](#)  
[method](#)— Variante à utiliser pour [la soumission de formulaire](#)  
[name](#)— Nom du formulaire à utiliser dans l' `document.forms` API  
[novalidate](#)— Contourner la validation du contrôle du formulaire pour [la soumission du formulaire](#)  
[target](#)— [Navigable](#) pour [la soumission du formulaire](#)  
[rel](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

[Exposed=Window,

[LegacyOverrideBuiltIns](#),

[LegacyUnenumerableNamedProperties](#)]

interface **HTMLFormElement** : [HTMLElement](#) {

[[HTMLConstructor](#)] constructor();

[[CEReactions](#)] attribute DOMString [acceptCharset](#);

[[CEReactions](#)] attribute USVString [action](#);

[[CEReactions](#)] attribute DOMString [autocomplete](#);

[[CEReactions](#)] attribute DOMString [enctype](#);

[[CEReactions](#)] attribute DOMString [encoding](#);

[[CEReactions](#)] attribute DOMString [method](#);

[[CEReactions](#)] attribute DOMString [name](#);

[[CEReactions](#)] attribute boolean [noValidate](#);

[[CEReactions](#)] attribute DOMString [target](#);

[[CEReactions](#)] attribute DOMString [rel](#);

[SameObject, PutForwards=[value](#)] readonly attribute

[DOMTokenList](#) [relList](#);

[SameObject] readonly attribute

[HTMLFormControlsCollection](#) [elements](#);

readonly attribute unsigned long [length](#);

[getter](#) [Element](#) (unsigned long index);

[getter](#) ([RadioNodeList](#) or [Element](#)) (DOMString name);

```

undefined submit\(\);

undefined requestSubmit(optional HTMLElement? submitter =
null);

[CEReactions] undefined reset\(\);

boolean checkValidity\(\);

boolean reportValidity\(\);

};

```

L' [form](#)élément [représente](#) un [lien hypertexte](#) qui peut être manipulé via une collection d' [éléments associés à un formulaire](#) , dont certains peuvent représenter des valeurs modifiables pouvant être soumises à un serveur pour traitement.

L' [accept-charset](#)attribut donne les encodages de caractères qui doivent être utilisés pour la soumission. Si spécifié, la valeur doit être une correspondance [ASCII non sensible à la casse](#) pour " UTF-8". [\[CODAGE\]](#)

L' [name](#)attribut représente le [form](#)nom du dans la [forms](#) collection. La valeur ne doit pas être la chaîne vide et la valeur doit être unique parmi les [form](#)éléments de la [forms](#)collection dans laquelle elle se trouve, le cas échéant.

L' [autocomplete](#)attribut est un [attribut énuméré](#) . L'attribut a deux états. Le [on](#)mot clé correspond à l' état **activé** et le [off](#)mot clé correspond à l' état **désactivé** . L'attribut peut également être omis. La [valeur par défaut manquante](#) et la [valeur par défaut non valide](#) correspondent à l' état [activé](#) . L' état [désactivé](#) indique que, par défaut, les contrôles de formulaire dans le formulaire auront leur [nom de champ de remplissage automatique](#) défini sur " [off](#)" ; l' état [activé](#) indique que, par défaut, les contrôles de formulaire dans le formulaire auront leur [nom de champ de remplissage automatique](#) défini sur " [on](#)".

Les attributs [action](#), [enctype](#), , et sont [method](#)des [attributs de soumission de formulaire](#) .[novalidate](#)[target](#)

L' [rel](#)attribut sur [form](#)les éléments contrôle les types de liens créés par les éléments. La valeur de l'attribut doit être un [ensemble non ordonné de jetons uniques séparés par des espaces](#) . Les [mots clés autorisés et leurs significations](#) sont définis dans une section précédente.

[rel](#)Les [jetons pris en charge par](#) sont les mots clés définis dans [les types de liens HTML](#) qui sont autorisés sur [form](#)les éléments, ont un impact sur le modèle de traitement et sont pris en charge par l'agent utilisateur. [Les jetons pris en charge](#) possibles sont [noreferrer](#), [noopener](#)et [opener](#). [rel](#)Les [jetons pris en](#)

[charge par](#) doivent uniquement inclure les jetons de cette liste pour lesquels l'agent utilisateur implémente le modèle de traitement.

#### `form.elements`

✓

Renvoie un [HTMLFormControlsCollection](#) des contrôles de formulaire dans le formulaire (à l'exception des boutons d'image pour des raisons historiques).

#### `form.length`

✓

Renvoie le nombre de champs de formulaire dans le formulaire (hors boutons image pour des raisons historiques).

#### `form[index]`

Renvoie l' *index* ème élément du formulaire (hors boutons image pour des raisons historiques).

#### `form[name]`

Renvoie le champ de formulaire (ou, s'il y en a plusieurs, un [RadioNodeList](#) des champs de formulaire) dans le formulaire avec l' [ID](#) donné ou [name](#) (hors boutons d'image pour des raisons historiques) ; ou, s'il n'y en a pas, renvoie l' [img](#) élément avec l'ID donné.

Une fois qu'un élément a été référencé à l'aide d'un nom particulier, ce nom continuera d'être disponible comme moyen de référencer cet élément dans cette méthode, même si l'ID réel de l'élément ou change, tant que l' élément [name](#) reste dans l' [arbre](#) .

S'il existe plusieurs éléments correspondants, un [RadioNodeList](#) objet contenant tous ces éléments est renvoyé.

#### `form.submit()`

✓

Soumet le formulaire, en contournant [la validation de contrainte interactive](#) et sans déclencher d' [submit](#) événement.

#### `form.requestSubmit([ submitter ])`

✓

Demande de soumettre le formulaire. Contrairement à [submit\(\)](#), cette méthode inclut [la validation interactive des contraintes](#) et le déclenchement d'un [submit](#) événement, chacun pouvant annuler la soumission.

L' argument *émetteur* peut être utilisé pour pointer vers un [bouton d'envoi](#) spécifique , , , et peuvent avoir un impact sur [formaction](#) la [formenctype](#) soumission . De plus, le soumissionnaire sera inclus lors [de la construction de la liste d'inscription](#) pour la soumission ; normalement, les boutons sont exclus. [formmethod](#) [formnovalidate](#) [formtarget](#)

`form.reset()`



Réinitialise le formulaire.

`form.checkValidity()`

Renvoie vrai si les contrôles du formulaire sont tous valides ; sinon, renvoie faux.

`form.reportValidity()`

Renvoie vrai si les contrôles du formulaire sont tous valides ; sinon, renvoie false et informe l'utilisateur.

L' `autocomplete` attribut IDL doit [réfléter](#) l'attribut de contenu du même nom, [limité aux seules valeurs connues](#) .



MDN

Les attributs `name` et `rel` IDL doivent [réfléter](#) l'attribut de contenu du même nom.



MDN

L' `acceptCharset` attribut IDL doit [réfléter](#) l' `accept-charset` attribut content.

L' `relList` attribut IDL doit [réfléter](#) l' `rel` attribut content.

---

L' `elements` attribut IDL doit renvoyer un `HTMLFormControlsCollection` enraciné à la `racine` `form` de l'élément , dont le filtre correspond [aux éléments répertoriés](#) dont [le propriétaire du formulaire](#) est l' élément, à l'exception des éléments dont l'attribut est à l' état `Image Button` , qui doivent, pour des raisons historiques, être exclus de cet élément particulier. collection. `forminputtype`

L' `length` attribut IDL doit renvoyer le nombre de nœuds [représentés](#) par la `elements` collection.

Les [indices de propriété pris en charge](#) à tout instant sont les indices pris en charge par l'objet renvoyé par l' `elements` attribut à cet instant.

Pour [déterminer la valeur d'une propriété indexée](#) pour un `form` élément, l'agent utilisateur doit renvoyer la valeur renvoyée par la `item` méthode sur la `elements` collection, lorsqu'elle est invoquée avec l'index donné comme argument.

---

Chaque formélément a un mappage de noms vers des éléments appelé la **carte des noms passés** . Il est utilisé pour conserver les noms des contrôles même lorsqu'ils changent de nom.

Les noms de propriété pris en charge se composent des noms obtenus à partir de l'algorithme suivant, dans l'ordre obtenu à partir de cet algorithme :

1. Soit *les noms sourcés* une liste ordonnée initialement vide de tuples composée d'une chaîne, d'un élément, d'une source, où la source est soit *id* , *name* , soit *past* , et, si la source est *past* , un âge.
2. Pour chaque élément candidat répertorié dont le propriétaire du formulaire est l' formélément, à l'exception de tous inputles éléments dont typel'attribut est à l' état Image Button :
  1. Si *candidate* a un idattribut, ajoutez une entrée aux *noms sourcés* avec id la valeur de cet attribut comme chaîne, *candidate* comme élément et *id* comme source.
  2. Si *le candidat* a un nameattribut, ajoutez une entrée aux *noms sourcés* avec namela valeur de cet attribut comme chaîne, *le candidat* comme élément et *le nom* comme source.
3. Pour chaque imgélément *candidat* dont le propriétaire du formulaire est l' formélément :
  1. Si *candidate* a un idattribut, ajoutez une entrée aux *noms sourcés* avec id la valeur de cet attribut comme chaîne, *candidate* comme élément et *id* comme source.
  2. Si *le candidat* a un nameattribut, ajoutez une entrée aux *noms sourcés* avec namela valeur de cet attribut comme chaîne, *le candidat* comme élément et *le nom* comme source.
4. Pour chaque entrée *passée* dans la carte des noms passés , ajoutez une entrée aux *noms sourcés* avec le nom de *l'entrée passée* comme chaîne, l'élément de *l'entrée passée* comme élément, *le passé* comme source et la durée pendant laquelle *l'entrée passée* a été dans le passé, les noms correspondent à l'âge.
5. Trier *les noms sourcés* par ordre arborescent de l'entrée d'élément de chaque tuple, trier les entrées avec le même élément en plaçant les entrées dont la source est *id* en premier, puis les entrées dont la source est *name* , et enfin les entrées dont la source est *past* , et trier les entrées avec le même élément et source par leur âge, le plus ancien en premier.

6. Supprimez toutes les entrées dans *les noms sourcés* qui ont la chaîne vide comme nom.
7. Supprimez toutes les entrées dans *les noms sourcés* qui ont le même nom qu'une entrée précédente dans la carte.
8. Renvoie la liste des noms à partir *des noms sourcés* , en conservant leur ordre relatif.

Pour déterminer la valeur d'un *nom* de propriété nommé pour un formélément, l'agent utilisateur doit exécuter les étapes suivantes :

1. Laissez *les candidats* être un objet actif RadioNodeList contenant tous les éléments répertoriés , dont le propriétaire du formulaire est l' formélément, qui ont soit un idattribut, soit un nameattribut égal à *name* , à l'exception des inputéléments dont typel'attribut est dans l' état Image Button , dans l' ordre de l'arborescence .
2. Si *candidates* est vide, laissez *candidates* être un objet actif RadioNodeList contenant tous les imgéléments, dont le propriétaire du formulaire est l' formélément, qui ont soit un idattribut, soit un nameattribut égal à *name* , dans l'ordre de l'arborescence .
3. Si *candidates* est vide, *nom* est le nom d'une des entrées dans la carte des anciens nomsform de l'élément : renvoie l'objet associé à *nom* dans cette carte.
4. Si *candidates* contient plus d'un nœud, renvoie *candidates* .
5. Sinon, *les candidats* contiennent exactement un nœud. Ajoutez un mappage du *nom* au nœud dans *les candidats* dans la carte des noms passésform de l'élément , en remplaçant l'entrée précédente par le même nom, le cas échéant.
6. Renvoie le nœud dans *les candidats* .

Si un élément répertorié dans la carte des anciens nomsform d'un élément change de propriétaire de formulaire , ses entrées doivent être supprimées de cette carte.

---

La submit() méthode, lorsqu'elle est invoquée, doit soumettre l' form élément à partir de l' formélément lui-même, avec l' indicateur de *méthode soumis* àsubmit() partir de défini.

La méthode, lorsqu'elle est invoquée, doit exécuter les étapes suivantes : `requestSubmit(submitter)`

1. Si l'*émetteur* n'est pas nul, alors :
  1. Si l'*émetteur* n'est pas un [bouton d'envoi](#) , lancez un `TypeError`.
  2. Si le propriétaire du [formulaire de l'expéditeur](#) n'est pas cet élément, lancez un `" ".formNotFoundError DOMException`
2. Sinon, définissez l'*émetteur* sur cet `form`élément.
3. [Soumettez](#) cet `form`élément, à partir de `submitter` .

La `reset()` méthode, lorsqu'elle est invoquée, doit exécuter les étapes suivantes :

1. Si l' `form`élément est marqué comme [verrouillé pour la réinitialisation](#) , alors revenez.
2. Marquez l' `form`élément comme **verrouillé pour la réinitialisation** .
3. [Réinitialisez](#) l' `form`élément.
4. Décochez l' `form`élément comme [verrouillé pour la réinitialisation](#) .

Si la `checkValidity()` méthode est invoquée, l'agent utilisateur doit [valider statiquement les contraintes](#) de l' `form`élément, et retourner true si la validation de la contrainte retourne un résultat *positif* , et false si elle retourne un résultat *négatif* .

Si la `reportValidity()` méthode est invoquée, l'agent utilisateur doit [valider interactivement les contraintes](#) de l' `form`élément, et retourner true si la validation de la contrainte retourne un résultat *positif* , et false si elle retourne un résultat *négatif* .

Cet exemple montre deux formulaires de recherche :

```
<form action="https://www.google.com/search" method="get">
  <label>Google: <input type="search" name="q"></label> <input
type="submit" value="Search...">
</form>

<form action="https://www.bing.com/search" method="get">
  <label>Bing: <input type="search" name="q"></label> <input
type="submit" value="Search...">
</form>
```

#### 4.10.4 L' `label`élément





## Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu interactif](#) .  
[Contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

## Modèle de contenu :

[Phrase content](#) , mais sans [éléments étiquetables](#) descendants sauf s'il s'agit du [contrôle étiqueté](#) `label` de l'élément, et sans éléments descendants .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

[Attributs globaux](#)  
`for`— Associer le libellé au contrôle de formulaire

## Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

## Interface DOM :

```
[Exposed=Window]

interface HTMLLabelElement : HTMLElement {

    [HTMLConstructor] constructor();

    readonly attribute HTMLFormElement? form;

    [CEReactions] attribute DOMString htmlFor;

    readonly attribute HTMLElement? control;

};
```

L' `label` élément [représente](#) une légende dans une interface utilisateur. La légende peut être associée à un contrôle de formulaire spécifique, appelé **contrôle étiqueté** `label` de l'élément , soit en utilisant l' attribut, soit en plaçant le contrôle de formulaire à l'intérieur de l' élément lui-même. [forlabel](#)

Sauf indication contraire dans les règles suivantes, un [label](#) élément n'a pas [de contrôle étiqueté](#) .



L' [for](#)attribut peut être spécifié pour indiquer un contrôle de formulaire auquel la légende doit être associée. Si l'attribut est spécifié, la valeur de l'attribut doit être l' [ID](#) d'un [élément étiquetable](#) dans la même [arborescence](#) que l' [label](#) élément. Si l'attribut est spécifié et qu'il existe un élément dans l' [arborescence](#) dont l'[ID](#) est égal à la valeur de l' [for](#)attribut, et que le premier élément de ce type dans l'[ordre de l'arborescence](#) est un [élément étiquetable](#) , alors cet élément est le [contrôle étiqueté](#) [label](#) de l'élément .

Si l' [for](#)attribut n'est pas spécifié, mais que l' [label](#) élément a un descendant [d'élément étiquetable](#) , le premier descendant de ce type dans l'[arborescence](#) est le [contrôle étiqueté](#) [label](#) de l'élément .

La [label](#) présentation et le comportement par défaut exacts de l'élément, en particulier son [comportement d'activation](#) , le cas échéant, doivent correspondre au comportement de l'étiquette de la plate-forme. Le [comportement d'activation](#) d'un [label](#) élément pour les événements ciblant les descendants [de contenu interactif](#) d'un [label](#) élément, et tous les descendants de ces descendants [de contenu interactif](#) , doit être de ne rien faire.

*[Les éléments personnalisés associés au formulaire sont des éléments étiquetables](#) , donc pour les agents utilisateurs où le [comportement d'activation](#) [label](#) de l'élément a un impact sur le [contrôle étiqueté](#) , les éléments intégrés et personnalisés seront impactés.*

Par exemple, sur les plates-formes où cliquer sur une étiquette active le contrôle du formulaire, cliquer sur [label](#) dans l'extrait de code suivant pourrait déclencher l'agent utilisateur pour [déclencher un click](#) événement sur l' [input](#) élément, comme si l'élément lui-même avait été déclenché par l'utilisateur :

```
<label><input type=checkbox name=lost> Lost</label>
```

De même, en supposant qu'il `my-checkbox` a été déclaré comme [élément personnalisé associé au formulaire](#) (comme dans [cet exemple](#) ), alors le code

```
<label><my-checkbox name=lost></my-checkbox> Lost</label>
```

aurait le même comportement, [déclenchant un click](#) événement sur l' `my-checkbox` élément.

Sur d'autres plates-formes, le comportement dans les deux cas peut être simplement de concentrer le contrôle ou de ne rien faire.

L'exemple suivant montre trois contrôles de formulaire chacun avec une étiquette, dont deux ont un petit texte indiquant le bon format à utiliser par les utilisateurs.

```
<p><label>Full name: <input name=fn> <small>Format: First
Last</small></label></p>

<p><label>Age: <input name=age type=number min=0></label></p>

<p><label>Post code: <input name=pc> <small>Format: AB12
3CD</small></label></p>
```

### **label.control**



Renvoie le contrôle de formulaire associé à cet élément.

### **label.form**



Renvoie le [propriétaire du formulaire](#) du contrôle de formulaire associé à cet élément.

Renvoie null s'il n'y en a pas.



MDN

L' **htmlFor** attribut IDL doit [refléter](#) l' **for** attribut content.

L' **control** attribut IDL doit renvoyer le [contrôle étiqueté](#) **label** de l'élément , le cas échéant, ou null s'il n'y en a pas.

L' **form** attribut IDL doit exécuter les étapes suivantes :

1. Si l' **label** élément n'a pas [de contrôle étiqueté](#) , renvoie null.
2. Si le [contrôle étiqueté](#) **label** de l'élément n'est pas un [élément associé à un formulaire](#) , renvoie null.
3. Renvoie le [propriétaire](#) du formulaire du [contrôle étiqueté](#) **label** de l'élément (qui peut toujours être nul).

*L' **form** attribut IDL sur l' **label** élément est différent de l' **form** attribut IDL sur [les éléments associés au formulaire répertoriés](#) et l' élément n'a pas d' attribut de contenu. [labelform](#)*

### **control.labels**



Renvoie un [NodeList](#) de tous les **label** éléments auxquels le contrôle de formulaire est associé.

[Les éléments étiquetables](#) et tous [input](#) les éléments ont un objet [actif](#) [NodeList](#) qui leur est associé qui représente la liste des [label](#) éléments, dans [l'ordre de l'arborescence](#), dont [le contrôle étiqueté](#) est l'élément en question. L' [labels](#) attribut IDL des [éléments étiquetables](#) qui ne sont pas [des éléments personnalisés associés au formulaire](#) et l' [labels](#) attribut IDL des [input](#) éléments, lors de l'obtention, doivent renvoyer cet [NodeList](#) objet, et cette même valeur doit toujours être renvoyée, à moins que cet élément ne soit un [input](#) élément dont [type](#) l'attribut est dans le État [masqué](#), auquel cas il doit à la place renvoyer null.

MDN

[Les éléments personnalisés associés au formulaire](#) n'ont pas d' [labels](#) attribut IDL. Au lieu de cela, leur [ElementInternals](#) objet a un [labels](#) attribut IDL. Lors de l'obtention, il doit lancer un ["NotSupportedError"](#) [DOMException](#) si l' [élément cible](#) n'est pas un [élément personnalisé associé au formulaire](#). Sinon, il doit renvoyer cet [NodeList](#) objet et cette même valeur doit toujours être renvoyée.

Cet exemple (non conforme) montre ce qui arrive à [NodeList](#) et ce qui [labels](#) revient quand un [input](#) élément voit son [type](#) attribut modifié.

```
<!doctype html>
<p><label><input></label></p>
<script>
  const input = document.querySelector('input');
  const labels = input.labels;
  console.assert(labels.length === 1);

  input.type = 'hidden';
  console.assert(labels.length === 0); // the input is no longer the
  label's labeled control
  console.assert(input.labels === null);

  input.type = 'checkbox';
  console.assert(labels.length === 1); // the input is once again
  the label's labeled control
  console.assert(input.labels === labels); // same value as returned
  originally
</script>
```

#### 4.10.5 L' [input](#) élément

✓ MDN

## Catégories :

[Contenu du flux](#) .

[Contenu de la phrase](#) .

Si l' [type](#)attribut n'est pas à l' état [Masqué](#) : [contenu interactif](#) .

Si l' [type](#)attribut n'est pas dans l'

état [Masqué](#) : [Listed](#) , [labelable](#) , [submittable](#) , [resettable](#) et [autocapitalize-inheriting form-associated element](#) .

Si l' [type](#)attribut est à l'

état [Masqué](#) : [Listed](#) , [submittable](#) , [resettable](#) et [autocapitalize-inheriting form-associated element](#) .

Si l' [type](#)attribut n'est pas à l' état [Masqué](#) : [contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

## Modèle de contenu :

[Rien](#) .

## Omission de balise dans text/html :

Pas [de balise de fin](#) .

## Attributs de contenu :

[Attributs globaux](#)

[accept](#)- Conseil pour le type de fichier attendu dans [les contrôles de téléchargement de fichiers](#)

[alt](#)— Texte de remplacement à utiliser lorsque les images ne sont pas disponibles

[autocomplete](#)- Astuce pour la fonction de remplissage automatique du formulaire

[checked](#)— Si le contrôle est coché

[dirname](#)— Nom du contrôle de formulaire à utiliser pour envoyer la [directionnalité](#) de l'élément lors de [la soumission du formulaire](#)

[disabled](#)— Si le contrôle du formulaire est désactivé

[form](#)— Associe l'élément à un [form](#)élément

[formaction](#)— [URL](#) à utiliser pour [la soumission du formulaire](#)

[formenctype](#)— Type d'encodage [de la liste d'entrées](#) à utiliser pour [la soumission du formulaire](#)

[formmethod](#)— Variante à utiliser pour [la soumission de formulaire](#)

[formnovalidate](#)— Contourner la validation du contrôle du formulaire pour [la soumission du formulaire](#)

[formtarget](#)— [Navigable](#) pour [la soumission du formulaire](#)

[height](#)— Dimension verticale

[list](#)— Liste des options de saisie semi-automatique

[max](#)- Valeur maximum

[maxlength](#)— [Longueur](#) maximale de la valeur

[min](#)- Valeur minimum

[minlength](#)— [Longueur](#) minimale de la valeur

multiple— S'il faut autoriser plusieurs valeurs  
name— Nom de l'élément à utiliser pour [la soumission du formulaire](#) et dans l' `form.elements` API  
pattern— Modèle à mettre en correspondance avec la valeur du contrôle de formulaire  
placeholder— Étiquette visible par l'utilisateur à placer dans le contrôle du formulaire  
popoverhidetarget  
popovershowtarget  
popovertoggletarget  
readonly— Autoriser ou non la modification de la valeur par l'utilisateur  
required— Si le contrôle est requis pour [la soumission du formulaire](#)  
size— Taille du contrôle  
src— Adresse de la ressource  
step— Granularité à faire correspondre à la valeur du contrôle de formulaire  
type— Type de contrôle de formulaire  
value— Valeur du champ de formulaire  
width— Dimensions horizontales  
De plus, l' title attribut [a une sémantique spéciale](#) sur cet élément : Description du modèle (lorsqu'il est utilisé avec pattern l'attribut).

### **Considérations d'accessibilité :**

type attribut à l' état [Caché](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut à l' état [Texte](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Recherche](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut à l' état [Téléphone](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [de l'URL](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Email](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut à l' état [Mot de passe](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Date](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut à l' état [Mois](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Semaine](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Temps](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Date et heure locales](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut à l' état [Numéro](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Range](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Couleur](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut à l' état [Case à cocher](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Bouton radio](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [File Upload](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [du bouton Soumettre](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Image Button](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Reset Button](#) : [pour les auteurs](#) ; [pour les exécutants](#) .  
type attribut dans l' état [Bouton](#) : [pour les auteurs](#) ; [pour les exécutants](#) .

## Interface DOM :

[Exposed=Window]

interface **HTMLInputElement** : [HTMLElement](#) {

[[HTMLConstructor](#)] constructor();

[[CEReactions](#)] attribute DOMString [accept](#);

[[CEReactions](#)] attribute DOMString [alt](#);

[[CEReactions](#)] attribute DOMString [autocomplete](#);

[[CEReactions](#)] attribute boolean [defaultChecked](#);

attribute boolean [checked](#);

[[CEReactions](#)] attribute DOMString [dirName](#);

[[CEReactions](#)] attribute boolean [disabled](#);

readonly attribute [HTMLFormElement](#)? [form](#);

attribute [FileList](#)? [files](#);

[[CEReactions](#)] attribute USVString [formAction](#);

[[CEReactions](#)] attribute DOMString [formEnctype](#);

[[CEReactions](#)] attribute DOMString [formMethod](#);

[[CEReactions](#)] attribute boolean [formNoValidate](#);

[[CEReactions](#)] attribute DOMString [formTarget](#);

[[CEReactions](#)] attribute unsigned long [height](#);

attribute boolean [indeterminate](#);

readonly attribute [HTMLDataListElement](#)? [list](#);

[[CEReactions](#)] attribute DOMString [max](#);

[[CEReactions](#)] attribute long [maxLength](#);

[[CEReactions](#)] attribute DOMString [min](#);

[[CEReactions](#)] attribute long [minLength](#);

[[CEReactions](#)] attribute boolean [multiple](#);

[[CEReactions](#)] attribute DOMString [name](#);

[[CEReactions](#)] attribute DOMString [pattern](#);

[[CEReactions](#)] attribute DOMString [placeholder](#);

[[CEReactions](#)] attribute boolean [readOnly](#);

[[CEReactions](#)] attribute boolean [required](#);

[[CEReactions](#)] attribute unsigned long [size](#);

[[CEReactions](#)] attribute USVString [src](#);

[[CEReactions](#)] attribute DOMString [step](#);

[[CEReactions](#)] attribute DOMString [type](#);

[[CEReactions](#)] attribute DOMString [defaultValue](#);

[[CEReactions](#)] attribute [[LegacyNullToEmptyString](#)]

DOMString [value](#);

attribute [object](#)? [valueAsDate](#);

attribute unrestricted double [valueAsNumber](#);

[[CEReactions](#)] attribute unsigned long [width](#);

undefined [stepUp](#)(optional long n = 1);

undefined [stepDown](#)(optional long n = 1);



```
readonly attribute boolean willValidate;
```

```
readonly attribute ValidityState validity;
```

```
readonly attribute DOMString validationMessage;
```

```
boolean checkValidity();
```

```
boolean reportValidity();
```

```
undefined setCustomValidity(DOMString error);
```

```
readonly attribute NodeList? labels;
```

```
undefined select();
```

```
attribute unsigned long? selectionStart;
```

```
attribute unsigned long? selectionEnd;
```

```
attribute DOMString? selectionDirection;
```

```
undefined setRangeText(DOMString replacement);
```

```
undefined setRangeText(DOMString replacement, unsigned
```

```
long start, unsigned long end, optional SelectionMode
```

```
selectionMode = "preserve");
```

```
undefined setSelectionRange(unsigned long start, unsigned
```

```
long end, optional DOMString direction);
```

```
undefined showPicker();
```

```
// also has obsolete members
```

```
};
```

```
HTMLInputElement includes PopoverTargetElement;
```

L' **input** élément [représente](#) un champ de données typé, généralement avec un contrôle de formulaire pour permettre à l'utilisateur de modifier les données.

L' **type** attribut contrôle le type de données (et le contrôle associé) de l'élément. C'est un [attribut énuméré](#) . Le tableau suivant répertorie les mots-clés et les états de l'attribut — les mots-clés de la colonne de gauche correspondent aux états de la cellule de la deuxième colonne sur la même ligne que le mot-clé.

Mot-clé	État	Type de données	Type de contrôle
<b>hidden</b>	<a href="#">Caché</a>	Une chaîne arbitraire	n / A
<b>text</b>	<a href="#">Texte</a>	Texte sans saut de ligne	Un contrôle de texte
<b>search</b>	<a href="#">Recherche</a>	Texte sans saut de ligne	Contrôle de la recherche
<b>tel</b>	<a href="#">Téléphone</a>	Texte sans saut de ligne	Un contrôle de texte
<b>url</b>	<a href="#">URL</a>	Une URL absolue	Un contrôle de texte
<b>email</b>	<a href="#">E-mail</a>	Une adresse e-mail ou une liste d'adresses e-mail	Un contrôle de texte
<b>password</b>	<a href="#">Mot de passe</a>	Texte sans saut de ligne (informations sensibles)	Un contrôle de texte qui masque la saisie de données
<b>date</b>	<a href="#">Date</a>	Une date (année, mois, jour) sans fuseau horaire	Un contrôle des dates
<b>month</b>	<a href="#">Mois</a>	Une date composée d'une année et d'un mois sans fuseau horaire	Un mois de contrôle
<b>week</b>	<a href="#">Semaine</a>	Une date composée d'un numéro semaine-année et d'un numéro de semaine sans fuseau horaire	Une semaine de contrôle
<b>time</b>	<a href="#">Temps</a>	Une heure (heure, minute, seconde, fraction de seconde) sans fuseau horaire	Un contrôle du temps
<b>datetime-local</b>	<a href="#">Date et heure locales</a>	Une date et une heure (année, mois, jour, heure, minute, seconde, fraction de seconde) sans fuseau horaire	Un contrôle de la date et de l'heure
<b>number</b>	<a href="#">Nombre</a>	Une valeur numérique	Un contrôle de texte ou un contrôle spinner

Mot-clé	État	Type de données	Type de contrôle
range	<a href="#">Gamme</a>	Une valeur numérique, avec la sémantique supplémentaire que la valeur exacte n'est pas importante	Un curseur ou similaire
color	<a href="#">Couleur</a>	Une couleur sRGB avec des composants rouge, vert et bleu 8 bits	Un sélecteur de couleurs
checkbox	<a href="#">Case à cocher</a>	Un ensemble de zéro ou plusieurs valeurs d'une liste prédéfinie	Une case à cocher
radio	<a href="#">Bouton radio</a>	Une valeur énumérée	Un bouton radio
file	<a href="#">Téléchargement de fichiers</a>	Zéro ou plusieurs fichiers chacun avec un <a href="#">type MIME</a> et éventuellement un nom de fichier	Une étiquette et un bouton
submit	<a href="#">Bouton de soumission</a>	Une valeur énumérée, avec la sémantique supplémentaire qu'il doit s'agir de la dernière valeur sélectionnée et initie la soumission du formulaire	Un bouton
image	<a href="#">Bouton d'image</a>	Une coordonnée, relative à la taille d'une image particulière, avec la sémantique supplémentaire qu'il doit s'agir de la dernière valeur sélectionnée et initie la soumission du formulaire	Soit une image cliquable, soit un bouton
reset	<a href="#">Bouton de réinitialisation</a>	n / A	Un bouton
button	<a href="#">Bouton</a>	n / A	Un bouton

La [valeur par défaut manquante](#) et la [valeur par défaut non valide](#) sont l' état [Texte](#) .

Lequel des attributs [accept](#), [alt](#), [autocomplete](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [list](#), [max](#), [maxlength](#), [min](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [readonly](#), [required](#), [size](#), [src](#), [step](#), et [width](#) content, les attributs [checked](#), [files](#), [valueAsDate](#), [valueAsNumber](#), et [list](#) IDL, la [select\(\)](#) méthode, les attributs [selectionStart](#), [selectionEnd](#) et [selectionDirection](#), IDL , les méthodes [setRangeText\(\)](#) et [setSelectionRange\(\)](#), les méthodes [stepUp\(\)](#) et [stepDown\(\)](#) et les [input](#) méthodes et [change](#) événements **s'appliquent** à un [input](#) élément dépend de l'état de son [type](#) attribut. Les sous-sections qui définissent chaque type définissent également clairement dans les sections normatives de "comptabilité" lesquelles de ces caractéristiques s'appliquent et lesquelles **ne s'appliquent pas** à chaque

type. Le comportement de ces fonctionnalités dépend de leur application ou non, tel que défini dans leurs différentes sections (qv pour [les attributs de contenu](#) , pour [les API](#) , pour [les événements](#) ).

Le tableau suivant est non normatif et résume lesquels de ces attributs de contenu, attributs IDL, méthodes et événements **s'appliquent** à chaque état :

	C a c h é	Text e ,Re cher che	URL ,Télé pho ne	E - m a i l	M o d e p a s s e	Date ,Mo is ,sema ine ,Tem ps	D a t e e t h e u r e l o c a l e s	N o m b r e	G a m m e	C o u l e u r	Cas e à coc her , Bout on radi o	Télé char gem ent de fichi ers	Bo uto n de so um issi on	B o u t o n d'i m a g e	Bouto n de réinitia lisation ,Bou ton
Attributs de contenu															
<u>accep t</u>	.	.	.	.	.	.	.	.	.	.	.	Oui	.	.	.
<u>alt</u>	.	.	.	.	.	.	.	.	.	.	.	.	.	O ui	.
<u>autoc omple te</u>	O ui	Oui	Oui	O ui	O ui	Oui	O ui	O ui	O ui	O ui	.	.	.	.	.
<u>check ed</u>	.	.	.	.	.	.	.	.	.	.	Oui	.	.	.	.
<u>dirna me</u>	.	Oui	.	.	.	.	.	.	.	.	.	.	.	.	.
<u>forma ction</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	O ui	.
<u>forme nctyp e</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	O ui	.
<u>formm ethod</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	O ui	.
<u>formn ovali date</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	O ui	.
<u>formt arget</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	O ui	.
<u>heigh t</u>	.	.	.	.	.	.	.	.	.	.	.	.	.	O ui	.

	<u>C</u> <u>a</u> <u>c</u> <u>h</u> <u>é</u>	<u>T</u> <u>e</u> <u>,</u> <u>R</u> <u>e</u> <u>c</u> <u>h</u> <u>e</u> <u>r</u> <u>c</u> <u>h</u> <u>e</u>	<u>U</u> <u>R</u> <u>L</u> <u>,</u> <u>T</u> <u>é</u> <u>l</u> <u>é</u> <u>p</u> <u>h</u> <u>o</u> <u>n</u> <u>e</u>	<u>E</u> <u>-</u> <u>m</u> <u>a</u> <u>i</u> <u>!</u>	<u>M</u> <u>o</u> <u>t</u> <u>d</u> <u>e</u> <u>p</u> <u>a</u> <u>s</u> <u>s</u> <u>e</u>	<u>D</u> <u>a</u> <u>t</u> <u>e</u> <u>,</u> <u>M</u> <u>o</u> <u>i</u> <u>s</u> <u>,</u> <u>s</u> <u>e</u> <u>m</u> <u>a</u> <u>i</u> <u>n</u> <u>e</u> <u>,</u> <u>T</u> <u>e</u> <u>m</u> <u>p</u> <u>s</u>	<u>D</u> <u>a</u> <u>t</u> <u>e</u> <u>e</u> <u>t</u> <u>h</u> <u>e</u> <u>u</u> <u>r</u> <u>e</u> <u>l</u> <u>c</u> <u>a</u> <u>l</u> <u>e</u> <u>s</u>	<u>N</u> <u>o</u> <u>m</u> <u>b</u> <u>r</u> <u>e</u>	<u>G</u> <u>a</u> <u>m</u> <u>m</u> <u>e</u>	<u>C</u> <u>o</u> <u>u</u> <u>l</u> <u>e</u> <u>u</u> <u>r</u>	<u>C</u> <u>a</u> <u>s</u> <u>e</u> <u>à</u> <u>c</u> <u>o</u> <u>c</u> <u>h</u> <u>e</u> <u>r</u> <u>,</u> <u>B</u> <u>o</u> <u>u</u> <u>t</u> <u>o</u> <u>n</u> <u>r</u> <u>a</u> <u>d</u> <u>i</u> <u>o</u>	<u>T</u> <u>é</u> <u>l</u> <u>é</u> <u>c</u> <u>h</u> <u>a</u> <u>r</u> <u>g</u> <u>e</u> <u>m</u> <u>e</u> <u>n</u> <u>t</u> <u>d</u> <u>e</u> <u>f</u> <u>i</u> <u>c</u> <u>h</u> <u>i</u> <u>e</u> <u>r</u> <u>s</u>	<u>B</u> <u>o</u> <u>u</u> <u>t</u> <u>o</u> <u>n</u> <u>d</u> <u>e</u> <u>s</u> <u>o</u> <u>m</u> <u>i</u> <u>s</u> <u>s</u> <u>i</u> <u>o</u> <u>n</u>	<u>B</u> <u>o</u> <u>u</u> <u>t</u> <u>o</u> <u>n</u> <u>d</u> <u>'</u> <u>i</u> <u>m</u> <u>a</u> <u>g</u> <u>e</u>	<u>B</u> <u>o</u> <u>u</u> <u>t</u> <u>o</u> <u>n</u> <u>d</u> <u>e</u> <u>r</u> <u>é</u> <u>i</u> <u>n</u> <u>i</u> <u>t</u> <u>i</u> <u>a</u> <u>t</u> <u>i</u> <u>o</u> <u>n</u> <u>,</u> <u>B</u> <u>o</u> <u>u</u> <u>t</u> <u>o</u> <u>n</u>
<u>list</u>	.	Oui	Oui	Oui	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.
<u>max</u>	.	.	.	.	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.
<u>maxle</u> <u>ngth</u>	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.	.	.	.	.
<u>min</u>	.	.	.	.	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.
<u>minle</u> <u>ngth</u>	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.	.	.	.	.
<u>multi</u> <u>ple</u>	.	.	.	Oui	.	.	.	.	.	.	.	Oui	.	.	.
<u>patte</u> <u>rn</u>	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.	.	.	.	.
<u>place</u> <u>holde</u> <u>r</u>	.	Oui	Oui	Oui	Oui	.	.	Oui	.	.	.	.	.	.	.
<u>popov</u> <u>erhid</u> <u>etarg</u> <u>et</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	Oui	Oui
<u>popov</u> <u>ersho</u> <u>wtarg</u> <u>et</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	Oui	Oui
<u>popov</u> <u>ertog</u> <u>gleta</u> <u>rget</u>	.	.	.	.	.	.	.	.	.	.	.	.	Oui	Oui	Oui
<u>reado</u> <u>nly</u>	.	Oui	Oui	Oui	Oui	Oui	Oui	Oui	.	.	.	.	.	.	.
<u>requi</u> <u>red</u>	.	Oui	Oui	Oui	Oui	Oui	Oui	Oui	.	.	Oui	Oui	.	.	.

[illegible]

	<a href="#">C</a> <a href="#">a</a> <a href="#">c</a> <a href="#">h</a> <a href="#">é</a>	<a href="#">Text</a> <a href="#">e</a> , <a href="#">Re</a> <a href="#">cher</a> <a href="#">che</a>	<a href="#">URL</a> <a href="#">,Télé</a> <a href="#">pho</a> <a href="#">ne</a>	<a href="#">E</a> : <a href="#">mai</a> <a href="#">!</a>	<a href="#">M</a> <a href="#">o</a> <a href="#">t</a> <a href="#">d</a> <a href="#">e</a> <a href="#">p</a> <a href="#">a</a> <a href="#">s</a> <a href="#">s</a> <a href="#">e</a>	<a href="#">Date</a> , <a href="#">Mo</a> <a href="#">is</a> , <a href="#">sema</a> <a href="#">ine</a> , <a href="#">Tem</a> <a href="#">ps</a>	<a href="#">D</a> <a href="#">a</a> <a href="#">t</a> <a href="#">e</a> <a href="#">et</a> <a href="#">h</a> <a href="#">e</a> <a href="#">ur</a> <a href="#">e</a> <a href="#">l</a> <a href="#">o</a> <a href="#">c</a> <a href="#">a</a> <a href="#">l</a> <a href="#">e</a> <a href="#">s</a>	<a href="#">N</a> <a href="#">o</a> <a href="#">m</a> <a href="#">b</a> <a href="#">r</a> <a href="#">e</a>	<a href="#">G</a> <a href="#">a</a> <a href="#">m</a> <a href="#">m</a> <a href="#">e</a>	<a href="#">C</a> <a href="#">o</a> <a href="#">u</a> <a href="#">l</a> <a href="#">e</a> <a href="#">u</a> <a href="#">r</a>	<a href="#">Cas</a> <a href="#">e</a> <a href="#">à</a> <a href="#">coc</a> <a href="#">her</a> , <a href="#">Bout</a> <a href="#">on</a> <a href="#">radi</a> <a href="#">o</a>	<a href="#">Télé</a> <a href="#">char</a> <a href="#">gem</a> <a href="#">ent</a> <a href="#">de</a> <a href="#">fichi</a> <a href="#">ers</a>	<a href="#">Bo</a> <a href="#">uto</a> <a href="#">n</a> <a href="#">de</a> <a href="#">so</a> <a href="#">um</a> <a href="#">issi</a> <a href="#">on</a>	<a href="#">B</a> <a href="#">o</a> <a href="#">ut</a> <a href="#">o</a> <a href="#">n</a> <a href="#">d'i</a> <a href="#">m</a> <a href="#">a</a> <a href="#">g</a> <a href="#">e</a>	<a href="#">Bouto</a> <a href="#">n</a> <a href="#">de</a> <a href="#">réinitia</a> <a href="#">lisatio</a> <a href="#">n</a> , <a href="#">Bou</a> <a href="#">ton</a>	
<a href="#">selec</a> <a href="#">tionE</a> <a href="#">nd</a>	.	Oui	Oui	.	Oui	.	.	.	.	.	.	.	.	.	.	.
<a href="#">selec</a> <a href="#">tionD</a> <a href="#">irect</a> <a href="#">ion</a>	.	Oui	Oui	.	Oui	.	.	.	.	.	.	.	.	.	.	.
<a href="#">setRa</a> <a href="#">ngeTe</a> <a href="#">xt()</a>	.	Oui	Oui	.	Oui	.	.	.	.	.	.	.	.	.	.	.
<a href="#">setSe</a> <a href="#">lecti</a> <a href="#">onRan</a> <a href="#">ge()</a>	.	Oui	Oui	.	Oui	.	.	.	.	.	.	.	.	.	.	.
<a href="#">stepD</a> <a href="#">own()</a>	.	.	.	.	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.	.
<a href="#">stepU</a> <a href="#">p()</a>	.	.	.	.	.	Oui	Oui	Oui	Oui	.	.	.	.	.	.	.
Événements																
<a href="#">input</a> <a href="#">é</a> <a href="#">vénem</a> <a href="#">ent</a>	.	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	.	.	.	.
<a href="#">chang</a> <a href="#">e</a> <a href="#">évène</a> <a href="#">ment</a>	.	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	Oui	.	.	.	.

† Si le contrôle n'a pas de texte sélectionnable, la [select\(\)](#) méthode aboutit à un no-op, sans "[InvalidStateError](#)" [DOMException](#).

Certains états de l' [type](#)attribut définissent un **algorithme de nettoyage de valeur**.

Chaque [input](#)élément a une [valeur](#), qui est exposée par l' [value](#)attribut IDL. Certains états définissent un **algorithme pour convertir une chaîne en nombre**, un **algorithme pour convertir un nombre en chaîne**, un **algorithme**

**pour convertir une chaîne en `Date`objet et un algorithme pour convertir un `Date`objet en chaîne** , qui sont utilisés

par `max`, `min`, `step`, `valueAsDate`, `valueAsNumber` et `stepUp()`.

L' [indicateur de valeur sale](#) d' un [input](#) élément doit être défini sur `true` chaque fois que l'utilisateur interagit avec le contrôle d'une manière qui modifie la [valeur](#) . (Il est également défini sur `true` lorsque la valeur est modifiée par programme, comme décrit dans la définition de l' attribut IDL.)[value](#)

L' [value](#) attribut content donne la [valeur](#) par défaut de l' [input](#) élément. Lorsque l' [value](#) attribut content est ajouté, défini ou supprimé, si l' [indicateur de valeur sale](#) du contrôle est faux, l'agent utilisateur doit définir la [valeur](#) de l'élément sur la valeur de l' [value](#) attribut content, s'il y en a un, ou la chaîne vide sinon, puis exécutez l' [algorithme de nettoyage de la valeur](#) actuelle , s'il est défini.

Chaque [input](#) élément a une [valeur de vérification](#) , qui est exposée par l' [checked](#) attribut IDL.

Chaque [input](#) élément a un **indicateur booléen de vérification sale** . Quand c'est vrai, on dit que l'élément a une **coche sale** . L' [indicateur de vérification sale](#) doit être initialement défini sur `false` lorsque l'élément est créé, et doit être défini sur `true` chaque fois que l'utilisateur interagit avec le contrôle d'une manière qui modifie la [vérification](#) .



L' [checked](#) attribut content est un [attribut booléen](#) qui donne la [vérification](#) par défaut de l' [input](#) élément. Lorsque l' [checked](#) attribut content est ajouté, si le contrôle n'a pas [de vérification sale](#) , l'agent utilisateur doit définir la [vérification](#) de l'élément sur `true` ; lorsque l' [checked](#) attribut content est supprimé, si le contrôle n'a pas [de vérification sale](#) , l'agent utilisateur doit définir la [vérification](#) de l'élément sur `false`.

L' [algorithme de réinitialisation](#) pour [input](#) les éléments consiste à définir l' [indicateur de valeur sale](#) et l'[indicateur de vérification sale](#) sur `false`, de définir la [valeur](#) de l'élément sur la valeur de l' [value](#) attribut de contenu, s'il y en a un, ou la chaîne vide sinon, de définir la [vérification](#) du element sur `true` si l'élément a un [checked](#) attribut content et `false` s'il n'en a pas, vide la liste des [fichiers sélectionnés](#) , puis invoque l' [algorithme de nettoyage des valeurs](#) , si l' [type](#) état actuel de l'attribut en définit un.

Chaque [input](#) élément peut être [modifiable](#) . Sauf indication contraire, un [input](#) élément est toujours [mutable](#) . De même, sauf indication contraire, l'agent utilisateur ne doit pas autoriser l'utilisateur à modifier la [valeur](#) ou la [vérification](#) de l'élément .

Lorsqu'un [input](#) élément est [désactivé](#) , il n'est pas [modifiable](#) .



*L' readonly attribut peut également dans certains cas (par exemple pour l' état Date , mais pas l' état Checkbox ) empêcher un input élément d'être mutable .*

Les étapes de clonage des input éléments doivent propager la valeur , l'indicateur de valeur sale , la vérification et l'indicateur de vérification sale du nœud cloné à la copie.

Le comportement d'activation des input éléments est le suivant :

1. Si cet élément n'est pas modifiable et n'est pas dans l' état Case à cocher et n'est pas dans l' état Radio , alors revenez.
2. Exécutez le **comportement d'activation d'entrée** de cet élément , le cas échéant, et ne faites rien d'autre.
3. Exécutez le comportement d'activation de l'attribut cible popover sur cet élément.

*Rappelez-vous que le comportement d'activation d'un élément s'exécute à la fois pour les activations initiées par l'utilisateur et pour les activations synthétiques (par exemple, via `el.click()`). Les agents utilisateurs peuvent également avoir des comportements pour un contrôle donné - non spécifiés ici - qui ne sont déclenchés que par de véritables activations initiées par l'utilisateur. Un choix courant consiste à afficher le sélecteur, le cas échéant , pour le contrôle. En revanche, le comportement d'activation d'entrée affiche uniquement des sélecteurs pour les cas historiques particuliers des états Téléchargement de fichier et Couleur .*

Le comportement de pré-activation hérité pour input les éléments est le suivant :

1. Si type l'attribut de cet élément est dans l' état Checkbox , alors définissez la vérification de cet élément sur sa valeur opposée (c'est-à-dire true s'il est false, false s'il est true) et définissez l' indeterminate attribut IDL de cet élément sur false.
2. Si l'attribut de cet élément type est dans l' état Bouton radio , obtenez une référence à l'élément du groupe de boutons radio de cet élément dont la vérification est définie sur true, le cas échéant, puis définissez la vérification de cet élément sur true.

Le comportement legacy-canceled-activation pour input les éléments est le suivant :

1. Si l'attribut de l'élément type est dans l' état Case à cocher , réinitialisez l' état coché de l'élément et l' indeterminate attribut IDL de l'élément sur les valeurs qu'ils avaient avant l' exécution du comportement de pré-activation hérité.
2. Si type l'attribut de cet élément est dans l' état Bouton radio , alors si l'élément auquel une référence a été obtenue dans le comportement hérité de pré-activation , le cas échéant, est toujours dans ce qui est maintenant le groupe

[de boutons radio](#) de cet élément , s'il en a encore un , et si c'est le cas, définir [la vérification](#) de cet élément sur true ; ou bien, s'il n'y avait pas un tel élément, ou si cet élément n'est plus dans le [groupe de boutons radio](#) de cet élément , ou si cet élément n'a plus de [groupe de boutons radio](#) , définir [la vérification](#) de cet élément sur false.

---

Lorsqu'un [input](#) élément est créé pour la première fois, le rendu et le comportement de l'élément doivent être définis sur le rendu et le comportement définis pour l' [type](#) état de l'attribut, et l' [algorithme de nettoyage de valeur](#) , s'il en existe un pour l' [type](#) état de l'attribut, doit être invoqué.

Lorsque l' attribut [input](#) d'un élément [type](#) change d'état, l'agent utilisateur doit exécuter les étapes suivantes :

1. Si l'état précédent de l'attribut de l'élément [type](#) met l' [value](#) attribut IDL en mode [valeur](#) , et que la [valeur](#) de l'élément n'est pas la chaîne vide, et que le nouvel état de l' [type](#) attribut de l'élément met l' [value](#) attribut IDL en mode [par défaut](#) ou en mode [par défaut/activé](#) , puis définissez l' [value](#) attribut content de l'élément sur la [valeur](#) de l'élément .
2. Sinon, si l'état précédent de l' [type](#) attribut de l'élément met l' [value](#) attribut IDL dans un mode autre que le mode [valeur](#) , et que le nouvel état de l'attribut de l'élément [type](#) met l' [value](#) attribut IDL dans le mode [valeur](#) , alors définissez la [valeur](#) de l'élément sur la valeur de l' [value](#) attribut content, s'il y en a un, ou la chaîne vide sinon, puis définissez l' [indicateur de valeur sale](#) du contrôle sur false.
3. Sinon, si l'état précédent de l' [type](#) attribut de l'élément a mis l' [value](#) attribut IDL dans un mode autre que le mode [nom de fichier](#) , et que le nouvel état de l' [type](#) attribut de l'élément met l' [value](#) attribut IDL dans le mode [nom de fichier](#) , alors définissez la [valeur](#) de l'élément sur vide chaîne.
4. Mettez à jour le rendu et le comportement de l'élément vers le nouvel état.
5. **Signale un changement de type** pour l'élément. (L' état [du bouton radio](#) utilise cela, en particulier.)
6. Appelez l' [algorithme de nettoyage de valeur](#) , s'il en existe un pour le [type](#) nouvel état de l'attribut.
7. Soit *previousSelectable* la valeur true si elle [setRangeText \(\)](#) a déjà [été appliquée](#) à l'élément, et false sinon.
8. Soit *nowSelectable* la valeur true si [setRangeText \(\)](#) maintenant [s'applique](#) à l'élément, et false sinon.

9. Si *previousSelectable* est false et *nowSelectable* est true, définissez la [position du curseur de saisie de texte](#) de l'élément au début du contrôle de texte et [définissez sa direction de sélection](#) sur " none".
- 

L' [name](#)attribut représente le nom de l'élément. L' attribut contrôle la façon dont la [directionnalité](#)[dirname](#) de l'élément est soumise. L' attribut est utilisé pour rendre le contrôle non interactif et pour empêcher la soumission de sa valeur. L' attribut est utilisé pour associer explicitement l' élément à son [propriétaire de formulaire](#) . L' attribut contrôle la façon dont l'agent utilisateur fournit le comportement de remplissage automatique. [disabledforminputautocomplete](#)



L' [indeterminate](#)attribut IDL doit initialement être défini sur false. Lors de l'obtention, il doit renvoyer la dernière valeur à laquelle il a été défini. Lors du réglage, il doit être réglé sur la nouvelle valeur. Cela n'a aucun effet, sauf pour modifier l'apparence des contrôles [de case à cocher](#) .



Les attributs [accept](#), [alt](#), [max](#), [min](#), [multiple](#), [pattern](#), [placeholder](#), [required](#), [size](#), [src](#) et IDL doivent [refléter](#) les attributs de contenu respectifs du même nom. L' attribut IDL doit [refléter](#) l' attribut content. L' attribut IDL doit [refléter](#) l' attribut content. L' attribut IDL doit [refléter](#) l' attribut content. L' attribut IDL doit [refléter](#) l' attribut content. [step](#)[dirname](#)[readonly](#)[readonly](#)[defaultChecked](#)[checked](#)[defaultValue](#)[value](#)

L' [type](#)attribut IDL doit [refléter](#) l'attribut de contenu respectif du même nom, [limité aux seules valeurs connues](#) . L' [maxLength](#)attribut IDL doit [refléter](#) l' [maxlength](#)attribut content, [limité aux seuls nombres non négatifs](#) . L' [minLength](#)attribut IDL doit [refléter](#) l' [minlength](#)attribut content, [limité aux seuls nombres non négatifs](#) .

Les attributs IDL [width](#) et [height](#) doivent renvoyer la largeur et la hauteur rendues de l'image, en [pixels CSS](#) , si une image est [rendue](#) et est rendue sur un support visuel ; ou bien la [largeur et la hauteur intrinsèques](#) de l'image, en [pixels CSS](#) , si une image est [disponible](#) mais n'est pas rendue sur un support visuel ; ou bien 0, si aucune image n'est [disponible](#) . Lorsque l' attribut [input](#) de l'élément [type](#) n'est pas dans l' état [Image Button](#) , aucune image n'est [disponible](#) . [\[CSS\]](#)

Lors de la définition, ils doivent agir comme s'ils [reflétaient](#) les attributs de contenu respectifs du même nom.

Les attributs [willValidate](#), [validity](#) et [validationMessage](#) IDL et les méthodes , et font partie de l' [API de validation](#) [decheckValidity\(\)](#) contrainte . L' attribut IDL fournit une liste des s de l'élément. Les méthodes , , , , et et les attributs IDL exposent la sélection de texte de l'élément. Les attributs , et IDL font partie de l'API des formulaires de l'élément. [reportValidity\(\)](#) [setCustomValidity\(\)](#) [labels](#) [labelselect\(\)](#) [selectionStart](#) [selectionEnd](#) [selectionDirection](#) [setRangeText\(\)](#) [setSelectionRange\(\)](#) [disabled](#) [formname](#)

#### 4.10.5.1 Etats de l' [type](#) attribut

##### 4.10.5.1.1 État masqué [type=hidden](#) ( )



Lorsque l'attribut [input](#) d'un élément [type](#) est à l' état [Masqué](#) , les règles de cette section s'appliquent.

L' [input](#) élément [représente](#) une valeur qui n'est pas destinée à être examinée ou manipulée par l'utilisateur.

**Validation de contrainte** : si l'attribut [input](#) d'un élément [type](#) est à l' état [Masqué](#) , il est [interdit de validation de contrainte](#) .

Si l' [name](#) attribut est présent et a une valeur qui est une correspondance [ASCII non sensible à la casse](#) pour " ", alors l' attribut [charset](#) de l'élément doit être omis. [value](#)

L' [autocomplete](#) attribut content [s'applique](#) à cet élément.

L' [value](#) attribut IDL [s'applique](#) à cet élément et est en mode [default](#) .

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [alt](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [list](#), [max](#), [maxlength](#), [min](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [popoverhidetarget](#), [popovershowtarget](#), [popovertoggletarget](#), [readonly](#), [required](#), [size](#), [src](#), [step](#) et [width](#).

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément : [checked](#), [files](#), [list](#), [selectionStart](#), [selectionEnd](#), [selectionDirection](#), [valueAsDate](#) et [valueAsNumber](#) IDL ; [select\(\)](#), [setRangeText\(\)](#), , et méthodes [setSelectionRange\(\)](#), [stepDown\(\)](#) [stepUp\(\)](#)

Les événements [input](#) et [ne s'appliquent pas](#) [change](#)

#### 4.10.5.1.2 État Texte ( `type=text` ) et État Recherche `type=search` ( )



Lorsque l'attribut `input` d'un élément `type` est à l'état [Texte](#) ou à l'état [Recherche](#), les règles de cette section s'appliquent.

L' `input` élément [représente](#) un contrôle d'édition de texte brut d'une ligne pour la [valeur](#) de l'élément .

*La différence entre l'état [Texte](#) et l'état [Recherche](#) est principalement stylistique : sur les plates-formes où les contrôles de recherche sont distingués des contrôles de texte normaux, l'état [Recherche](#) peut donner une apparence cohérente avec les contrôles de recherche de la plate-forme plutôt que d'apparaître comme un contrôle de texte normal.*

Si l'élément est [mutable](#), sa [valeur](#) doit être modifiable par l'utilisateur. Les agents utilisateurs ne doivent pas autoriser les utilisateurs à insérer les caractères U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR) dans la [valeur](#) de l'élément .

Si l'élément est [mutable](#), l'agent utilisateur doit permettre à l'utilisateur de changer le sens d'écriture de l'élément, en le définissant soit dans un sens d'écriture de gauche à droite, soit dans un sens d'écriture de droite à gauche. Si l'utilisateur le fait, l'agent utilisateur doit alors exécuter les étapes suivantes :

1. Définissez l' `dir` attribut de l'élément sur "`ltr`" si l'utilisateur a sélectionné un sens d'écriture de gauche à droite, et sur "`rtl`" s'il a sélectionné un sens d'écriture de droite à gauche.
2. [Mettez en file d'attente une tâche d'élément](#) sur la [source de tâche d'interaction utilisateur](#) donnée à l'élément pour [déclencher un événement](#) nommé `input` au niveau de l'élément, avec les attributs `bubbles` et `composed` initialisés sur `true`.

L' `value` attribut, s'il est spécifié, doit avoir une valeur qui ne contient aucun caractère U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR).

L' [algorithme de nettoyage de valeur](#) est le suivant : [Supprimez les sauts de ligne](#) de la [valeur](#) .

Les attributs de contenu d'élément communs `input`, les attributs IDL et les méthodes suivants [s'appliquent](#) à l'élément : `autocomplete`, `dirname`, `list`, `maxlength`, `minlength`, `pattern`, `placeholder`, `readonly`, `required` et `size` les attributs de contenu ; `list` les attributs , `selectionStart` , et IDL `selectionEnd` ; , , et méthodes `selectionDirection` `value` `select()` `setRangeText()` `setSelectionRange()`

L' `value` attribut IDL est en mode [value](#) .

Les événements [input](#) et [s'appliquent .change](#)

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [alt](#), [checked](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [max](#), [min](#), [multiple](#), [popover](#), [rhidetarget](#), [popovershowtarget](#), [popovertoggletarget](#), [src](#), [step](#) et [width](#).

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément : attributs [checked](#), [files](#), [valueAsDate](#) et [valueAsNumber](#) IDL ; [stepDown\(\)](#) et [stepUp\(\)](#) méthodes.

#### 4.10.5.1.3 État du téléphone `type=tel` ( )



MDN

Lorsque l'attribut [input](#) d'un élément [type](#) est à l'état [Téléphone](#), les règles de cette section s'appliquent.

L' [input](#) élément [représente](#) un contrôle permettant d'éditer un numéro de téléphone donné dans la [valeur](#) de l'élément .

Si l'élément est [mutable](#), sa [valeur](#) doit être modifiable par l'utilisateur. Les agents utilisateurs peuvent modifier l'espacement et, avec précaution, la ponctuation des [valeurs](#) saisies par l'utilisateur. Les agents utilisateurs ne doivent pas autoriser les utilisateurs à insérer les caractères U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR) dans la [valeur](#) de l'élément .

L' [value](#) attribut, s'il est spécifié, doit avoir une valeur qui ne contient aucun caractère U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR).

L' [algorithme de nettoyage de valeur](#) est le suivant : [Supprimez les sauts de ligne](#) de la [valeur](#) .

*Contrairement aux types [URL](#) et [E-mail](#), le type [Téléphone](#) n'applique pas de syntaxe particulière. C'est intentionnel ; en pratique, les champs de numéro de téléphone ont tendance à être des champs de forme libre, car il existe une grande variété de numéros de téléphone valides. Les systèmes qui doivent appliquer un format particulier sont encouragés à utiliser l' [pattern](#) attribut ou la [setCustomValidity\(\)](#) méthode pour se connecter au mécanisme de validation côté client.*

Les attributs de contenu d'élément communs [input](#), les attributs IDL et les méthodes suivants [s'appliquent](#) à l'élément : [autocomplete](#), [list](#), [maxlength](#), [minlength](#), [pattern](#), [placeholder](#), [readonly](#), [required](#) et [size](#) les attributs de contenu ; [list](#) les attributs , [selectionStart](#), , et IDL [selectionEnd](#) ; , , et méthodes [selectionDirection](#) [value](#) [select\(\)](#) [setRangeText\(\)](#) [setSelectionRange\(\)](#)

L' value attribut IDL est en mode value .

Les événements input et s'appliquent .change

Les attributs de contenu suivants ne doivent pas être spécifiés et ne s'appliquent pas à l'élément : accept, alt, checked, dirname, formaction, formenctype, formmethod, formnovalidate, formtarget, height, max, min, multiple, popoverhidetarget, popovershowtarget, popovertoggletarget, src, step et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément : attributs checked, files, valueAsDate et valueAsNumber IDL ; stepDown() et stepUp() méthodes.

#### 4.10.5.1.4 État de l'URL ( type=url )



Lorsque l'attribut input d'un élément type est à l' état URL , les règles de cette section s'appliquent.

L' input élément représente un contrôle pour éditer une seule URL absolue donnée dans la valeur de l'élément .

Si l'élément est modifiable , l'agent utilisateur doit autoriser l'utilisateur à modifier l'URL représentée par sa valeur . Les agents utilisateurs peuvent autoriser l'utilisateur à définir la valeur sur une chaîne qui n'est pas une URL absolue valide , mais peuvent également ou à la place automatiquement échapper les caractères entrés par l'utilisateur afin que la valeur soit toujours une URL absolue valide (même si ce n'est pas le cas la valeur réelle vue et éditée par l'utilisateur dans l'interface). Les agents utilisateurs doivent permettre à l'utilisateur de définir la valeur sur la chaîne vide. Les agents utilisateurs ne doivent pas autoriser les utilisateurs à insérer les caractères U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR) dans la valeur .

L' value attribut, s'il est spécifié et non vide, doit avoir une valeur qui est une URL valide potentiellement entourée d'espaces qui est également une URL absolue .

L' algorithme de nettoyage de valeur est le suivant : Supprimez les retours à la ligne de la valeur , puis supprimez les espaces blancs ASCII de début et de fin de la valeur .

**Validation de contrainte** : Alors que la valeur de l'élément n'est ni la chaîne vide ni une URL absolue valide , l'élément souffre d'une incompatibilité de type .

Les attributs de contenu d'élément communs input, les attributs IDL et les méthodes suivants s'appliquent à l'élément : autocomplete, list, maxlength, minlength, pattern, placeholder, readonly, required et size les attributs de contenu ; list les attributs



, `selectionStart`, , et IDL `selectionEnd`; , ,  
et méthodes `selectionDirection` `value` `select()` `setRangeText()` `setSelectionRange()`

L' `value` attribut IDL est en mode `value` .

Les événements `input` et `s'appliquent` `change`

Les attributs de contenu suivants ne doivent pas être spécifiés et `ne s'appliquent pas` à l'élément : `accept`, `alt`, `checked`, `dirname`, `formaction`, `formenctype`, `formmethod`, `formnovalidate`, `formtarget`, `height`, `max`, `min`, `multiple`, `popoverhidetarget`,  
, `povertarget`, `povertoggletarget`, `src`, `step` et `width`.

Les attributs et méthodes IDL suivants `ne s'appliquent pas` à l'élément :  
attributs `checked`, `files`, `valueAsDate` et `valueAsNumber` IDL  
; `stepDown()` et `stepUp()` méthodes.

Si un document contient le balisage suivant :

```
<input type="url" name="location" list="urls">
<datalist id="urls">
  <option label="MIME: Format of Internet Message Bodies"
value="https://www.rfc-editor.org/rfc/rfc2045">
  <option label="HTML" value="https://html.spec.whatwg.org/">
  <option label="DOM" value="https://dom.spec.whatwg.org/">
  <option label="Fullscreen"
value="https://fullscreen.spec.whatwg.org/">
  <option label="Media Session"
value="https://mediasession.spec.whatwg.org/">
  <option label="The Single UNIX Specification, Version 3"
value="http://www.unix.org/version3/">
</datalist>
```

... et l'utilisateur avait tapé " spec.w", et l'agent utilisateur avait également trouvé que l'utilisateur avait visité `https://url.spec.whatwg.org/#url-parsing` et `https://streams.spec.whatwg.org/` dans un passé récent, alors le rendu pourrait ressembler à ceci :

Les quatre premières URL de cet exemple sont constituées des quatre URL de la liste spécifiée par l'auteur qui correspondent au texte saisi par l'utilisateur, triées d'une manière définie par l'implémentation (peut-être en fonction de la fréquence à laquelle l' `utilisateur` se réfère à ces URL). Notez comment l'UA utilise la



connaissance que les valeurs sont des URL pour permettre à l'utilisateur d'omettre la partie schéma et d'effectuer une correspondance intelligente sur le nom de domaine.

Les deux dernières URL (et probablement bien d'autres, étant donné les indications de la barre de défilement indiquant que davantage de valeurs sont disponibles) sont les correspondances des données d'historique de session de l'agent utilisateur. Ces données ne sont pas mises à disposition de la page DOM. Dans ce cas particulier, l'UA n'a pas de titres à fournir pour ces valeurs.

#### 4.10.5.1.5 État de l'e-mail<sub>type=email</sub> ( )



Lorsque l'attribut input d'un élément type est à l'état E-mail, les règles de cette section s'appliquent.

Le fonctionnement de l'état E-mail varie selon que l'multiple attribut est spécifié ou non.

##### Lorsque l'multiple attribut n'est pas spécifié sur l'élément

L'input élément représente un contrôle pour modifier une adresse e-mail donnée dans la valeur de l'élément .

Si l'élément est modifiable, l'agent utilisateur doit autoriser l'utilisateur à modifier l'adresse e-mail représentée par sa valeur. Les agents utilisateurs peuvent autoriser l'utilisateur à définir la valeur sur une chaîne qui n'est pas une adresse e-mail valide. L'agent utilisateur doit agir d'une manière consistante à attendre de l'utilisateur qu'il fournisse une seule adresse e-mail. Les agents utilisateurs doivent permettre à l'utilisateur de définir la valeur sur la chaîne vide. Les agents utilisateurs ne doivent pas autoriser les utilisateurs à insérer les caractères U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR) dans la valeur. Les agents utilisateurs peuvent transformer la valeur pour l'affichage et l'édition ; en particulier, les agents utilisateurs devraient convertir le punycode dans les étiquettes de domaine de la valeur en IDN dans l'affichage et vice versa.

**Validation de contrainte** : alors que l'interface utilisateur représente une entrée que l'agent utilisateur ne peut pas convertir en punycode, le contrôle souffre d'une mauvaise entrée.

L'value attribut, s'il est spécifié et non vide, doit avoir une valeur correspondant à une seule adresse e-mail valide.

L'algorithme de nettoyage de valeur est le suivant : Supprimez les retours à la ligne de la valeur, puis supprimez les espaces blancs ASCII de début et de fin de la valeur.

**Validation de contrainte** : Alors que la [valeur](#) de l'élément n'est ni la chaîne vide ni une seule [adresse e-mail valide](#) , l'élément [souffre d'une incompatibilité de type](#) .

**Lorsque l' [multiple](#) attribut est spécifié sur l'élément**

L' [input](#) élément [représente](#) un contrôle pour ajouter, supprimer et modifier les adresses e-mail indiquées dans la [valeur s](#) de l'élément .

Si l'élément est [mutable](#) , l'agent utilisateur doit permettre à l'utilisateur d'ajouter, de supprimer et de modifier les adresses e-mail représentées par ses [valeurs](#) . Les agents utilisateurs peuvent autoriser l'utilisateur à définir n'importe quelle valeur individuelle dans la liste de [valeurs](#) sur une chaîne qui n'est pas une [adresse e-mail valide](#) , *mais* ne doivent pas autoriser les utilisateurs à définir une valeur individuelle sur une chaîne contenant U+002C COMMA (,), caractères U+000A SAUT DE LIGNE (LF) ou U+000D RETOUR CHARIOT (CR). Les agents utilisateurs doivent permettre à l'utilisateur de supprimer toutes les adresses dans les [valeurs](#) de l'élément . Les agents utilisateurs peuvent transformer les [valeurs](#) pour l'affichage et l'édition ; en particulier, les agents utilisateurs devraient convertir le punycode dans les étiquettes de domaine de la [valeur](#) en IDN dans l'affichage et vice versa.

**Validation de contrainte** : alors que l'interface utilisateur décrit une situation dans laquelle une valeur individuelle contient une virgule U+002C (,) ou représente une entrée que l'agent utilisateur ne peut pas convertir en punycode, le contrôle souffre d'une mauvaise [entrée](#) .

Chaque fois que l'utilisateur modifie la [valeur de l'élément s](#) , l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit *les dernières valeurs* une copie de la [valeur s](#) de l'élément .
2. [Supprimez les espaces blancs ASCII de début et de fin](#) de chaque valeur dans *les dernières valeurs* .
3. Laissez la [valeur](#) de l'élément être le résultat de la concaténation de toutes les valeurs dans *latest values* , en séparant chaque valeur de la suivante par un seul caractère U+002C COMMA (,), en maintenant l'ordre de la liste.

L' [value](#) attribut, s'il est spécifié, doit avoir une valeur correspondant à une [liste d'adresses e-mail valide](#) .

**L' [algorithme de nettoyage des valeurs](#) est le suivant :**

1. [Divisez par des virgules la valeur](#) de l'élément , [supprimez les espaces blancs ASCII de début et de fin](#) de chaque jeton résultant, le cas échéant, et laissez les [valeurs](#) de l'élément être la liste résultante (éventuellement vide) de jetons (éventuellement vides), en conservant l'ordre d'origine.

2. Laissez la [valeur](#) de l'élément être le résultat de la concaténation des [valeurs](#) de l'élément , en séparant chaque valeur de la suivante par un seul caractère U+002C COMMA (,), en maintenant l'ordre de la liste.

**Validation de contrainte** : alors que la [valeur](#) de l'élément n'est pas une [liste d'adresses e-mail valide](#) , l'élément [souffre d'une incompatibilité de type](#) .

Lorsque l' [multiple](#) attribut est défini ou supprimé, l'agent utilisateur doit exécuter l' [algorithme de nettoyage des valeurs](#) .

Une **adresse e-mail valide** est une chaîne qui correspond à la `email` production de l'ABNF suivant, dont le jeu de caractères est Unicode. Cet ABNF implémente les extensions décrites dans la RFC 1123. [\[ABNF\]](#) [\[RFC5322\]](#) [\[RFC1034\]](#) [\[RFC1123\]](#)

```
email      = 1*( atext / "." ) "@" label *( "." label )
label      = let-dig [ [ ldh-str ] let-dig ] ; limited to a
length of 63 characters by RFC 1034 section 3.5
atext      = < as defined in RFC 5322 section 3.2.3 >
let-dig    = < as defined in RFC 1034 section 3.5 >
ldh-str    = < as defined in RFC 1034 section 3.5 >
```

*Cette exigence est une [violation délibérée](#) de la RFC 5322, qui définit une syntaxe pour les adresses e-mail qui est à la fois trop stricte (avant le caractère "@"), trop vague (après le caractère "@") et trop laxiste (permettant des commentaires, des espaces caractères et chaînes entre guillemets d'une manière peu familière à la plupart des utilisateurs) pour être d'une utilité pratique ici. L'expression régulière compatible JavaScript et Perl suivante est une implémentation de la définition ci-dessus.*

```
/^[a-zA-Z0-9.!#$%&'*\+\-/?^_`{|}~-]+@[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?(?:\.[a-zA-Z0-9](?:[a-zA-Z0-9-]{0,61}[a-zA-Z0-9])?)*$/
```

Une **liste d'adresses e-mail valides** est un [ensemble de jetons séparés par des virgules](#) , où chaque jeton est lui-même une [adresse e-mail valide](#) . Pour obtenir la liste des jetons à partir d'une [liste d'adresses électroniques valides](#) , une implémentation doit [diviser la chaîne par des virgules](#) .

Les attributs de contenu d'élément communs [input](#), les attributs IDL et les méthodes suivants [s'appliquent](#) à l'élément : [autocomplete](#), [list](#), [maxlength](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [readonly](#), [required](#) et [size](#) les attributs de contenu ; [list](#) et [value](#) les attributs IDL ; [select\(\)](#) méthode.

L' [value](#) attribut IDL est en mode [value](#) .

Les événements [input](#) et [s'appliquent](#) [change](#)

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [alt](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [max](#), [min](#), [popover](#)

rhidetaget, popovershowtarget, popovertoggletarget, src,  
step et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément : checked, files, selectionStart, selectionEnd, selectionDirection, valueAsDate et valueAsNumber IDL ; setRangeText(), setSelectionRange(), stepDown() et stepUp() méthodes.

#### 4.10.5.1.6 État du mot de passe `type=password()`



Lorsque l'attribut input d'un élément type est à l'état Mot de passe, les règles de cette section s'appliquent.

L' input élément représente un contrôle d'édition de texte brut d'une ligne pour la valeur de l'élément. L'agent utilisateur doit masquer la valeur afin que les personnes autres que l'utilisateur ne puissent pas la voir.

Si l'élément est mutable, sa valeur doit être modifiable par l'utilisateur. Les agents utilisateurs ne doivent pas autoriser les utilisateurs à insérer les caractères U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR) dans la valeur.

L' value attribut, s'il est spécifié, doit avoir une valeur qui ne contient aucun caractère U+000A LINE FEED (LF) ou U+000D CARRIAGE RETURN (CR).

L' algorithme de nettoyage de valeur est le suivant : Supprimez les sauts de ligne de la valeur.

Les attributs de contenu d'élément communs input, les attributs IDL et les méthodes suivants s'appliquent à l'élément : autocomplete, maxlength, minlength, pattern, placeholder, readonly, required et size les attributs de contenu ; selectionStart les attributs, selectionEnd, selectionDirection et value IDL ; select(), setRangeText(), et setSelectionRange() méthodes.

L' value attribut IDL est en mode value.

Les événements input et s'appliquent change

Les attributs de contenu suivants ne doivent pas être spécifiés et ne s'appliquent pas à l'élément : accept, alt, checked, dirname, formaction, formenctype, formmethod, formnovalidate, formtarget, height, list, max, min, multiple, popoverhidetaget, popovershowtarget, popovertoggletarget, src, step et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément : attributs, checked, files et IDL ; et méthodes. listvalueAsDatevalueAsNumberstepDown() stepUp()

#### 4.10.5.1.7 État de la date ( `type=date` )



Lorsque l'attribut `input` d'un élément `type` est à l'état [Date](#) , les règles de cette section s'appliquent.

L' `input` élément [représente](#) un contrôle permettant de définir la [valeur](#) de l'élément sur une chaîne représentant une [date](#) spécifique .

Si l'élément est [mutable](#) , l'agent utilisateur doit permettre à l'utilisateur de modifier la [date](#) représentée par sa [valeur](#) , telle qu'obtenue en [analysant une date](#) à partir de celui-ci. Les agents utilisateurs ne doivent pas permettre à l'utilisateur de définir la [valeur](#) sur une chaîne non vide qui n'est pas une [chaîne de date valide](#) . Si l'agent utilisateur fournit une interface utilisateur pour sélectionner une [date](#) , la [valeur](#) doit être définie sur une [chaîne de date valide](#) représentant la sélection de l'utilisateur. Les agents utilisateurs doivent permettre à l'utilisateur de définir la [valeur](#) sur la chaîne vide.

**Validation de contrainte** : alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en une [chaîne de date valide](#) , le contrôle [souffre d'une mauvaise entrée](#) .

*Voir la [section d'introduction](#) pour une discussion sur la différence entre le format d'entrée et le format de soumission pour les contrôles de formulaire de date, d'heure et de nombre, et les [notes d'implémentation](#) concernant la localisation des contrôles de formulaire.*

L' `value` attribut, s'il est spécifié et non vide, doit avoir une valeur qui est une [chaîne de date valide](#) .

L' [algorithme de nettoyage de la valeur](#) est le suivant : si la [valeur](#) de l'élément n'est pas une [chaîne de date valide](#) , définissez-la à la place sur la chaîne vide.

L' `min` attribut, s'il est spécifié, doit avoir une valeur qui est une [chaîne de date valide](#) . L' `max` attribut, s'il est spécifié, doit avoir une valeur qui est une [chaîne de date valide](#) .

L' `step` attribut est exprimé en jours. Le [facteur d'échelle de pas](#) est de 86 400 000 (qui convertit les jours en millisecondes, comme utilisé dans les autres algorithmes). L' [étape par défaut](#) est de 1 jour.

Lorsque l'élément [souffre d'une discordance d'étape](#) , l'agent utilisateur peut arrondir la [valeur](#) de l'élément à la [date](#) la plus proche pour laquelle l'élément ne [souffrirait pas d'une discordance d'étape](#) .

L' [algorithme pour convertir une chaîne en nombre](#) , étant donné une *entrée de chaîne* , est le suivant : Si l'[analyse d'une date](#) à partir de l'entrée entraîne une erreur, alors renvoie une erreur ; sinon, renvoie le nombre de millisecondes écoulées

entre minuit UTC le matin du 1970-01-01 (l'heure représentée par la valeur " 1970-01-01T00:00:00.0Z) et minuit UTC le matin de la [date](#) analysée , en ignorant les secondes intercalaires.

L' [algorithme pour convertir un nombre en chaîne](#) , étant donné une **entrée numérique** , est le suivant : Renvoie une [chaîne de date valide](#) qui représente la [date](#) qui, en UTC, est l'*entrée* actuelle en millisecondes après minuit UTC le matin du 1970-01-01 ( le temps représenté par la valeur " 1970-01-01T00:00:00.0Z").

L' [algorithme pour convertir une chaîne en \*\*Date\*\*objet](#) , étant donné une **entrée de chaîne** , est le suivant : Si l'*analyse d'une date* à partir de l'*entrée* entraîne une erreur, alors renvoie une erreur ; sinon, renvoie [un nouvel \*\*Date\*\*objet](#) représentant minuit UTC le matin de la [date](#) analysée .

L' [algorithme pour convertir un \*\*Date\*\*objet en chaîne](#) , étant donné une **entrée **Date** d'objet** , est le suivant : Renvoie une [chaîne de date valide](#) qui représente la [date](#) actuelle à l'heure représentée par l'*entrée* dans le fuseau horaire UTC.

*L' état [Date](#) (et les autres états liés à la date et à l'heure décrits dans les sections suivantes) n'est pas destiné à la saisie de valeurs pour lesquelles une date et une heure précises par rapport au calendrier contemporain ne peuvent pas être établies. Par exemple, il serait inapproprié d'entrer des temps comme "une milliseconde après le big bang", "le début de la période jurassique", ou "un hiver vers 250 avant notre ère".*

*Pour la saisie de dates avant l'introduction du calendrier grégorien, les auteurs sont encouragés à ne pas utiliser l' état [Date](#) (et les autres états liés à la date et à l'heure décrits dans les sections suivantes), car les agents utilisateurs ne sont pas tenus de prendre en charge la conversion des dates et des périodes antérieures au calendrier grégorien, et demander aux utilisateurs de le faire manuellement impose une charge excessive aux utilisateurs. (Ceci est compliqué par la manière dont le calendrier grégorien a été mis en place, qui s'est produit à différents moments dans différents pays, allant de la moitié du XVI<sup>e</sup> siècle jusqu'au début du XX<sup>e</sup>.) Au lieu de cela, les auteurs sont encouragés à fournir des -contrôles d'entrée granulés utilisant l' [select](#) élément et [input](#) les éléments avec l' état [numérique](#) .*

Les attributs de contenu d'élément communs [input](#), les attributs IDL et les méthodes suivants [s'appliquent](#) à l'élément : [autocomplete](#), [list](#), [max](#), [min](#), [readonly](#), [required](#) et [step](#) les attributs de contenu ; [list](#) les attributs , [value](#), [valueAsDate](#) et [valueAsNumber](#) IDL ; [select\(\)](#), [stepDown\(\)](#), et [stepUp\(\)](#) méthodes.

L' [value](#) attribut IDL est en mode [value](#) .

Les événements [input](#) et [s'appliquent](#) [.change](#)

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [alt](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [maxlength](#),



, [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [popoverhidetarget](#), [popovershowtarget](#), [popovertoggletarget](#), [size](#), [src](#) et [width](#).

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément :

attributs [checked](#), [selectionStart](#), [selectionEnd](#) et [selectionDirection](#) IDL ; [setRangeText\(\)](#), et [setSelectionRange\(\)](#) méthodes.

#### 4.10.5.1.8 État du mois `type=month()`



Lorsque l'attribut [input](#) d'un élément [type](#) est à l'état [Mois](#), les règles de cette section s'appliquent.

L' [input](#) élément [représente](#) un contrôle pour définir la [valeur de l'élément sur une chaîne représentant un mois](#) spécifique.

Si l'élément est [mutable](#), l'agent utilisateur devrait permettre à l'utilisateur de changer le [mois](#) représenté par sa [valeur](#), tel qu'obtenu en [analysant un mois](#) à partir de celui-ci. Les agents utilisateurs ne doivent pas permettre à l'utilisateur de définir la [valeur](#) sur une chaîne non vide qui n'est pas une [chaîne de mois valide](#). Si l'agent utilisateur fournit une interface utilisateur pour sélectionner un [mois](#), alors la [valeur](#) doit être définie sur une [chaîne de mois valide](#) représentant la sélection de l'utilisateur. Les agents utilisateurs doivent permettre à l'utilisateur de définir la [valeur](#) sur la chaîne vide.

**Validation de contrainte** : Alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en une [chaîne de mois valide](#), le contrôle [souffre d'une mauvaise entrée](#).

*Voir la [section d'introduction](#) pour une discussion sur la différence entre le format d'entrée et le format de soumission pour les contrôles de formulaire de date, d'heure et de nombre, et les [notes d'implémentation](#) concernant la localisation des contrôles de formulaire.*

L' [value](#) attribut, s'il est spécifié et non vide, doit avoir une valeur correspondant à une [chaîne de mois valide](#).

L' [algorithme de nettoyage des valeurs](#) est le suivant : si la [valeur](#) de l'élément n'est pas une [chaîne de mois valide](#), définissez-la à la place sur la chaîne vide.

L' [min](#) attribut, s'il est spécifié, doit avoir une valeur correspondant à une [chaîne de mois valide](#). L' [max](#) attribut, s'il est spécifié, doit avoir une valeur correspondant à une [chaîne de mois valide](#).

L' step attribut est exprimé en mois. Le facteur d'échelle de l'étape est de 1 (aucune conversion n'est nécessaire car les algorithmes utilisent des mois). L' étape par défaut est de 1 mois.

Lorsque l'élément souffre d'une discordance d'étape , l'agent utilisateur peut arrondir la valeur de l'élément au mois le plus proche pour lequel l'élément ne souffrirait pas d'une discordance d'étape .

L' algorithme pour convertir une chaîne en nombre , étant donné une entrée de chaîne , est le suivant : Si l'analyse d'un mois à partir de l'entrée génère une erreur, alors renvoie une erreur ; sinon, renvoie le nombre de mois entre janvier 1970 et le mois analysé .

L' algorithme pour convertir un nombre en chaîne , étant donné une entrée numérique , est le suivant : Renvoie une chaîne de mois valide qui représente le mois qui a des mois d'entrée entre lui et janvier 1970.

L' algorithme pour convertir une chaîne en Dateobjet , étant donné une entrée de chaîne , est le suivant : Si l'analyse d'un mois à partir de l'entrée génère une erreur, alors renvoie une erreur ; sinon, renvoie un nouvel Dateobjet représentant minuit UTC le matin du premier jour du mois analysé .

L' algorithme pour convertir un Dateobjet en chaîne , étant donné une entrée Date d'objet , est le suivant : Renvoie une chaîne de mois valide qui représente le mois en cours à l'heure représentée par l'entrée dans le fuseau horaire UTC.

Les attributs de contenu d'élément communs input, les attributs IDL et les méthodes suivants s'appliquent à l'élément : autocomplete, list, max, min, readonly, required et step les attributs de contenu ; list les attributs , value, valueAsDate et valueAsNumber IDL ; select(), stepDown(), et stepUp() méthodes.

L' value attribut IDL est en mode value .

Les événements input et s'appliquent .change

Les attributs de contenu suivants ne doivent pas être spécifiés et ne s'appliquent pas à l'élément : accept, alt, checked, dirname, formaction, formenctype, formmethod, formnovalidate, formtarget, height, maxlength, minlength, multiple, pattern, placeholder, popoverhidetarget, popovershowtarget, popovertoggletarget, size, src et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément : attributs , checked, files et IDL ; , et méthodes selectionStart selectionEnd selectionDirection setRangeText() setSelectionRange()

#### 4.10.5.1.9 État de la semaine ( type=week )



Lorsque l'attribut `input` d'un élément `type` est à l'état [Semaine](#), les règles de cette section s'appliquent.

L'attribut `input` représente un contrôle permettant de définir la [valeur de l'élément sur une chaîne représentant une semaine](#) spécifique.

Si l'élément est [mutable](#), l'agent utilisateur devrait permettre à l'utilisateur de changer la [semaine](#) représentée par sa [valeur](#), telle qu'obtenue en [analysant une semaine](#) à partir de celui-ci. Les agents utilisateurs ne doivent pas autoriser l'utilisateur à définir la [valeur](#) sur une chaîne non vide qui n'est pas une [chaîne de semaine valide](#). Si l'agent utilisateur fournit une interface utilisateur pour sélectionner une [semaine](#), alors la [valeur](#) doit être définie sur une [chaîne de semaine valide](#) représentant la sélection de l'utilisateur. Les agents utilisateurs doivent permettre à l'utilisateur de définir la [valeur](#) sur la chaîne vide.

**Validation de contrainte** : alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en une [chaîne de semaine valide](#), le contrôle [souffre d'une mauvaise entrée](#).

*Voir la [section d'introduction](#) pour une discussion sur la différence entre le format d'entrée et le format de soumission pour les contrôles de formulaire de date, d'heure et de nombre, et les [notes d'implémentation](#) concernant la localisation des contrôles de formulaire.*

L'attribut `value`, s'il est spécifié et non vide, doit avoir une valeur correspondant à une [chaîne de semaine valide](#).

L'**[algorithme de nettoyage de la valeur](#)** est le suivant : si la [valeur](#) de l'élément n'est pas une [chaîne de semaine valide](#), définissez-la à la place sur la chaîne vide.

L'attribut `min`, s'il est spécifié, doit avoir une valeur correspondant à une [chaîne de semaine valide](#). L'attribut `max`, s'il est spécifié, doit avoir une valeur correspondant à une [chaîne de semaine valide](#).

L'attribut `step` est exprimé en semaines. Le [facteur d'échelle de pas](#) est de 604 800 000 (qui convertit les semaines en millisecondes, comme utilisé dans les autres algorithmes). L'[étape par défaut](#) est de 1 semaine. La [base de pas par défaut](#) est -259 200 000 (le début de la semaine 1970-S01).

Lorsque l'élément [souffre d'une discordance d'étape](#), l'agent utilisateur peut arrondir la [valeur](#) de l'élément à la [semaine](#) la plus proche pour laquelle l'élément ne [souffrirait pas d'une discordance d'étape](#).

L'**[algorithme pour convertir une chaîne en nombre](#)**, étant donné une *entrée de chaîne*, est le suivant : Si l'[analyse d'une chaîne de semaine](#) à partir de l'entrée entraîne une erreur, alors renvoie une erreur ; sinon, renvoie le nombre de millisecondes écoulées entre minuit UTC le matin du 1970-01-01 (l'heure

représentée par la valeur " " 1970-01-01T00:00:00.0Z) et minuit UTC le matin du lundi de la [semaine](#) analysée , en ignorant les secondes intercalaires.

L' [algorithme pour convertir un nombre en chaîne](#) , étant donné une **entrée** numérique , est le suivant : Renvoie une [chaîne de semaine valide](#) qui représente la [semaine](#) qui, en UTC, correspond à l'entrée actuelle en millisecondes après minuit UTC le matin du 1970-01-01 ( le temps représenté par la valeur " 1970-01-01T00:00:00.0Z").

L' [algorithme pour convertir une chaîne en Date objet](#) , étant donné une **entrée** de chaîne , est le suivant : Si l'[analyse d'une semaine](#) à partir de l'entrée génère une erreur, alors renvoie une erreur ; sinon, renvoie un [nouvel Date objet](#) représentant minuit UTC le matin du lundi de la [semaine](#) analysée .

L' [algorithme pour convertir un Date objet en chaîne](#) , étant donné une **entrée** [Date](#) d'objet , est le suivant : Renvoie une [chaîne de semaine valide](#) qui représente la [semaine](#) en cours à l'heure représentée par l'entrée dans le fuseau horaire UTC.

Les attributs de contenu d'élément communs [input](#), les attributs IDL et les méthodes suivants [s'appliquent](#) à l'élément : [autocomplete](#), [list](#), [max](#), [min](#), [readonly](#), [required](#) et [step](#) les attributs de contenu ; [list](#) les attributs , [value](#), [valueAsDate](#) et [valueAsNumber](#) IDL ; [select\(\)](#), [stepDown\(\)](#), et [stepUp\(\)](#) méthodes.

L' [value](#) attribut IDL est en mode [value](#) .

Les événements [input](#) et [s'appliquent .change](#)

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [alt](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [maxlength](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [popoverhidetarget](#), [popovershowtarget](#), [popovertoggletarget](#), [size](#), [src](#) et [width](#).

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément : attributs , [checked](#), [files](#) et IDL ; , et méthodes. [selectionStart](#) [selectionEnd](#) [selectionDirection](#) [setRangeText\(\)](#) [setSelectionRange\(\)](#)

#### 4.10.5.1.10 Etat horaire ( type=time)



Lorsque l'attribut [input](#) d'un élément [type](#) est dans l' état [Time](#) , les règles de cette section s'appliquent.

L' input élément représente un contrôle permettant de définir la valeur de l'élément sur une chaîne représentant une heure spécifique .

Si l'élément est mutable , l'agent utilisateur doit permettre à l'utilisateur de modifier l' heure représentée par sa valeur , telle qu'obtenue en analysant une heure à partir de celui-ci. Les agents utilisateurs ne doivent pas autoriser l'utilisateur à définir la valeur sur une chaîne non vide qui n'est pas une chaîne d'heure valide . Si l'agent utilisateur fournit une interface utilisateur pour sélectionner une heure , alors la valeur doit être définie sur une chaîne d'heure valide représentant la sélection de l'utilisateur. Les agents utilisateurs doivent permettre à l'utilisateur de définir la valeur sur la chaîne vide.

**Validation de contrainte** : alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en une chaîne de temps valide , le contrôle souffre d'une mauvaise entrée .

*Voir la section d'introduction pour une discussion sur la différence entre le format d'entrée et le format de soumission pour les contrôles de formulaire de date, d'heure et de nombre, et les notes d'implémentation concernant la localisation des contrôles de formulaire.*

L' value attribut, s'il est spécifié et non vide, doit avoir une valeur qui est une chaîne d'heure valide .

L' algorithme de nettoyage des valeurs est le suivant : si la valeur de l'élément n'est pas une chaîne de temps valide , définissez-la à la place sur la chaîne vide.

Le contrôle de formulaire a un domaine périodique .

L' min attribut, s'il est spécifié, doit avoir une valeur correspondant à une chaîne d'heure valide . L' max attribut, s'il est spécifié, doit avoir une valeur correspondant à une chaîne d'heure valide .

L' step attribut est exprimé en secondes. Le facteur d'échelle de pas est de 1000 (qui convertit les secondes en millisecondes, comme utilisé dans les autres algorithmes). Le pas par défaut est de 60 secondes.

Lorsque l'élément souffre d'une discordance d'étape , l'agent utilisateur peut arrondir la valeur de l'élément à l' instant le plus proche pour lequel l'élément ne souffrirait pas d'une discordance d'étape .

L' algorithme pour convertir une chaîne en nombre , étant donné une entrée de chaîne , est le suivant : Si l'analyse d'une heure à partir de l'entrée entraîne une erreur, alors renvoie une erreur ; sinon, renvoie le nombre de millisecondes écoulées entre minuit et l' heure analysée d'un jour sans changement d'heure.

L' algorithme pour convertir un nombre en chaîne , étant donné une entrée numérique , est le suivant : Renvoie une chaîne d'heure valide qui

représente l' [heure](#) saisie en *millisecondes* après minuit un jour sans changement d'heure.

L' [algorithme pour convertir une chaîne en \*\*Date\*\*objet](#) , étant donné une **entrée de chaîne** , est le suivant : Si l'[analyse d'une heure](#) à partir de l'entrée entraîne une erreur, alors renvoie une erreur ; sinon, renvoie un [nouvel \*\*Date\*\*objet](#) représentant l' [heure](#) analysée en UTC le 1970-01-01.

L' [algorithme pour convertir un \*\*Date\*\*objet en chaîne](#) , étant donné une **entrée **Date** d'objet** , est le suivant : Renvoie une [chaîne d'heure valide](#) qui représente le composant [de temps](#) UTC représenté par l'entrée .

Les attributs de contenu d'élément communs [input](#), les attributs IDL et les méthodes suivants [s'appliquent](#) à l'élément : [autocomplete](#), [list](#), [max](#), [min](#), [readonly](#), [required](#) et [step](#) les attributs de contenu ; [list](#) les attributs , [value](#), [valueAsDate](#) et [valueAsNumber](#) IDL ; [select\(\)](#) , [stepDown\(\)](#) , et [stepUp\(\)](#) méthodes.

L' [value](#) attribut IDL est en mode [value](#) .

Les événements [input](#) et [s'appliquent .change](#)

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [alt](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [maxlength](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [popoverhidetarget](#), [popovershowtarget](#), [popovertoggletarget](#), [size](#), [src](#) et [width](#).

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément : attributs , [checked](#), [files](#) et IDL ; , et méthodes [selectionStart](#) [selectionEnd](#) [selectionDirection](#) [setRangeText\(\)](#) [setSelectionRange\(\)](#)

#### 4.10.5.1.11 État de la date et de l'heure locales `type=datetime-local()`



Lorsque l' attribut [input](#) d'un élément [type](#) est dans l' état [Date et heure locales](#) , les règles de cette section s'appliquent.

L' [input](#) élément [représente](#) un contrôle permettant de définir la [valeur](#) de l'élément sur une chaîne représentant une [date et une heure locales](#) , sans informations de décalage de fuseau horaire.

Si l'élément est [mutable](#) , l'agent utilisateur doit permettre à l'utilisateur de modifier la [date et l'heure](#) représentées par sa [valeur](#) , telles qu'obtenues en [analysant une date et une heure](#) à partir de celui-ci. Les agents utilisateurs ne doivent pas autoriser l'utilisateur à définir la [valeur](#) sur une chaîne non vide qui n'est pas une [chaîne de](#)

[date et d'heure locale normalisée valide](#) . Si l'agent utilisateur fournit une interface utilisateur pour sélectionner une [date et une heure locales](#) , la [valeur](#) doit être définie sur une [chaîne de date et d'heure locale normalisée valide](#) représentant la sélection de l'utilisateur. Les agents utilisateurs doivent permettre à l'utilisateur de définir la [valeur](#) sur la chaîne vide.

**Validation de contrainte** : alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en une [chaîne de date et d'heure locale normalisée valide](#) , le contrôle [souffre d'une mauvaise entrée](#) .

*Voir la [section d'introduction](#) pour une discussion sur la différence entre le format d'entrée et le format de soumission pour les contrôles de formulaire de date, d'heure et de nombre, et les [notes d'implémentation](#) concernant la localisation des contrôles de formulaire.*

L' [value](#) attribut, s'il est spécifié et non vide, doit avoir une valeur correspondant à une [chaîne de date et d'heure locale valide](#) .

L' [algorithme de nettoyage des valeurs](#) est le suivant : si la [valeur](#) de l'élément est une [chaîne de date et d'heure locale valide](#) , définissez-la sur une [chaîne de date et d'heure locale normalisée valide](#) représentant la même date et heure ; sinon, définissez-le sur la chaîne vide à la place.

L' [min](#) attribut, s'il est spécifié, doit avoir une valeur correspondant à une [chaîne de date et d'heure locale valide](#) . L' [max](#) attribut, s'il est spécifié, doit avoir une valeur correspondant à une [chaîne de date et d'heure locale valide](#) .

L' [step](#) attribut est exprimé en secondes. Le [facteur d'échelle de pas](#) est de 1000 (qui convertit les secondes en millisecondes, comme utilisé dans les autres algorithmes). Le [pas par défaut](#) est de 60 secondes.

Lorsque l'élément [souffre d'une discordance d'étape](#) , l'agent utilisateur peut arrondir la [valeur](#) de l'élément à la [date et à l'heure locales](#) les plus proches pour lesquelles l'élément ne [souffrirait pas d'une discordance d'étape](#) .

L' [algorithme pour convertir une chaîne en nombre](#) , étant donné une **entrée de chaîne** , est le suivant : Si l'[analyse d'une date et d'une heure](#) à partir de l'entrée entraîne une erreur, alors renvoie une erreur ; sinon, renvoie le nombre de millisecondes écoulées depuis minuit le matin du 1970-01-01 (l'heure représentée par la valeur " 1970-01-01T00:00:00.0") jusqu'à la [date et l'heure locales](#) analysées , en ignorant les secondes intercalaires.

L' [algorithme pour convertir un nombre en chaîne](#) , étant donné une **entrée numérique** , est le suivant : Renvoie une [chaîne de date et d'heure locale normalisée valide](#) qui représente la date et l'heure entrées millisecondes après minuit le matin du 1970-01-01 ( le temps représenté par la valeur " 1970-01-01T00:00:00.0").

*Voir la [note sur les dates historiques](#) dans la section [État de la date](#) .*

Les attributs de contenu d'élément communs [input](#), les attributs IDL et les méthodes suivants [s'appliquent](#) à l'élément : [autocomplete](#), [list](#), [max](#), [min](#), [readonly](#), [required](#) et [step](#) les attributs de contenu ; [list](#), [value](#) et [valueAsNumber](#) attributs IDL ; [select\(\)](#), [stepDown\(\)](#), et [stepUp\(\)](#) méthodes.

L' [value](#) attribut IDL est en mode [value](#) .

Les événements [input](#) et [s'appliquent](#) [.change](#)

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [alt](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [maxlength](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [popoverhidetarget](#), [popovershowtarget](#), [popovertoggletarget](#), [size](#), [src](#) et [width](#).

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément : attributs [checked](#), [files](#), , , et IDL [selectionStart](#); , et méthodes [selectionEnd](#) [selectionDirection](#) [valueAsDate](#) [setRangeText\(\)](#) [setSelectionRange\(\)](#)

L'exemple suivant montre une partie d'une application de réservation de vol. L'application utilise un [input](#) élément dont [type](#) l'attribut est défini sur [datetime-local](#), puis interprète la date et l'heure données dans le fuseau horaire de l'aéroport sélectionné.

```
<fieldset>
  <legend>Destination</legend>
  <p><label>Airport: <input type=text name=to
list=airports></label></p>
  <p><label>Departure time: <input type=datetime-local name=totime
step=3600></label></p>
</fieldset>
<datalist id=airports>
  <option value=ATL label="Atlanta">
  <option value=MEM label="Memphis">
  <option value=LHR label="London Heathrow">
  <option value=LAX label="Los Angeles">
  <option value=FRA label="Frankfurt">
</datalist>
```

#### 4.10.5.1.12 Etat du nombre ( `type=number` )



Lorsque l'attribut [input](#) d'un élément [type](#) est à l'état [Nombre](#) , les règles de cette section s'appliquent.



L' inputélément représente une commande pour définir la valeur de l'élément sur une chaîne représentant un nombre.

Si l'élément est mutable , l'agent utilisateur doit permettre à l'utilisateur de modifier le nombre représenté par sa valeur , tel qu'obtenu en appliquant les  règles d'analyse des valeurs de nombres à virgule flottante  . Les agents utilisateurs ne doivent pas autoriser l'utilisateur à définir la valeur sur une chaîne non vide qui n'est pas un nombre à virgule flottante valide . Si l'agent utilisateur fournit une interface utilisateur pour sélectionner un nombre, alors la valeur doit être définie sur la meilleure représentation du nombre représentant la sélection de l'utilisateur sous la forme d'un nombre à virgule flottante . Les agents utilisateurs doivent permettre à l'utilisateur de définir la valeur sur la chaîne vide.

**Validation des contraintes** : alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en un nombre à virgule flottante valide , le contrôle souffre d'une mauvaise entrée .

*Cette spécification ne définit pas quelle interface utilisateur les agents utilisateurs doivent utiliser ; les vendeurs d'agents utilisateurs sont encouragés à considérer ce qui répondrait le mieux aux besoins de leurs utilisateurs. Par exemple, un agent utilisateur sur les marchés persan ou arabe peut prendre en charge la saisie numérique en persan et en arabe (en la convertissant au format requis pour la soumission comme décrit ci-dessus). De même, un agent utilisateur conçu pour Romans pourrait afficher la valeur en chiffres romains plutôt qu'en décimal ; ou (de manière plus réaliste) un agent utilisateur conçu pour le marché français pourrait afficher la valeur avec des apostrophes entre les milliers et les virgules avant les décimales, et permettre à l'utilisateur d'entrer une valeur de cette manière, en la convertissant en interne au format de soumission décrit ci-dessus.*

L' valueattribut, s'il est spécifié et non vide, doit avoir une valeur qui est un nombre à virgule flottante valide .

L' algorithme de nettoyage de valeur est le suivant : si la valeur de l'élément n'est pas un nombre à virgule flottante valide , définissez-le sur la chaîne vide à la place.

L' minattribut, s'il est spécifié, doit avoir une valeur qui est un nombre à virgule flottante valide . L' maxattribut, s'il est spécifié, doit avoir une valeur qui est un nombre à virgule flottante valide .

Le facteur d'échelle de pas est 1. Le pas par défaut est 1 (ce qui permet à l'utilisateur de ne sélectionner que des nombres entiers, sauf si la base de pas a une valeur non entière).

Lorsque l'élément souffre d'une discordance d'étape , l'agent utilisateur peut arrondir la valeur de l'élément au nombre le plus proche pour lequel l'élément ne souffrirait pas d'une discordance d'étape . S'il y a deux de ces nombres, les agents utilisateurs sont encouragés à choisir celui qui est le plus proche de l'infini positif.

L' **algorithme pour convertir une chaîne en nombre** , étant donné une **entrée de chaîne** , est le suivant : Si l'application des **règles d'analyse des valeurs de nombres à virgule flottante** aux résultats d' **entrée** génère une erreur, alors renvoie une erreur ; sinon, renvoie le nombre résultant.

L' **algorithme pour convertir un nombre en chaîne** , étant donné une **entrée numérique** , est le suivant : Renvoie un **nombre à virgule flottante valide** qui représente l'entrée .

Les attributs de contenu d'élément communs **input**, les attributs IDL et les méthodes suivants **s'appliquent** à l'élément : **autocomplete**, **list**, **max**, **min**, **placeholder**, **readonly**, **required** et **step** les attributs de contenu ; **list**, **value** et **valueAsNumber** attributs IDL ; **select()**, **stepDown()**, et **stepUp()** méthodes.

L' **value** attribut IDL est en mode **value** .

Les événements **input** et **s'appliquent** **change**

Les attributs de contenu suivants ne doivent pas être spécifiés et **ne s'appliquent pas** à l'élément : **accept**, **alt**, **checked**, **dirname**, **formaction**, **formenctype**, **formmethod**, **formnovalidate**, **formtarget**, **height**, **maxlength**, **minlength**, **multiple**, **pattern**, **popoverhidetarget**, **popovershowtarget**, **popovertoggletarget**, **size**, **src** et **width**.

Les attributs et méthodes IDL suivants **ne s'appliquent pas** à l'élément : attributs **checked**, **files**, , , et IDL **selectionStart**; , et méthodes **selectionEnd**, **selectionDirection**, **valueAsDate**, **setRangeText()** **setSelectionRange()**

Voici un exemple d'utilisation d'un contrôle d'entrée numérique :

```
<label>How much do you want to charge? $<input type=number min=0 step=0.01 name=price></label>
```

Comme décrit ci-dessus, un agent utilisateur peut prendre en charge l'entrée numérique dans le format local de l'utilisateur, en le convertissant au format requis pour la soumission comme décrit ci-dessus. Cela peut inclure la gestion des séparateurs de groupement (comme dans "872 000 000 000") et divers séparateurs décimaux (tels que "3,99" contre "3.99") ou l'utilisation de chiffres locaux (tels que ceux en arabe, devanagari, persan et thaï).

*L' **type=number** état n'est pas approprié pour une entrée qui se compose uniquement de nombres mais qui n'est pas à proprement parler un nombre. Par exemple, il serait inapproprié pour les numéros de carte de crédit ou les codes postaux américains. Une façon simple de déterminer s'il faut l'utiliser **type=number** est de considérer s'il serait logique que le contrôle d'entrée ait une interface spinbox (par exemple avec des flèches "haut" et "bas"). Obtenir un numéro de carte de crédit erroné de 1 dans le dernier chiffre n'est pas une erreur mineure, c'est aussi faux que d'avoir chaque chiffre incorrect. Il n'aurait donc pas de sens pour l'utilisateur de sélectionner un numéro de carte de crédit à l'aide des boutons "haut" et "bas". Lorsqu'une interface spinbox n'est pas appropriée, **type=text** c'est*



*probablement le bon choix (éventuellement avec un attribut inputmode ou pattern ).*

#### 4.10.5.1.13 État de la plage ( `type=range` )



Lorsque l'attribut input d'un élément type est dans l'état Plage , les règles de cette section s'appliquent.

L' input élément représente un contrôle pour définir la valeur de l'élément sur une chaîne représentant un nombre, mais avec la mise en garde que la valeur exacte n'est pas importante, laissant les UA fournir une interface plus simple qu'ils ne le font pour l'état Nombre .

Si l'élément est mutable , l'agent utilisateur doit permettre à l'utilisateur de modifier le nombre représenté par sa valeur , tel qu'obtenu en appliquant les règles d'analyse des valeurs de nombres à virgule flottante . Les agents utilisateurs ne doivent pas permettre à l'utilisateur de définir la valeur sur une chaîne qui n'est pas un nombre à virgule flottante valide . Si l'agent utilisateur fournit une interface utilisateur pour sélectionner un nombre, alors la valeur doit être définie sur une meilleure représentation du nombre représentant la sélection de l'utilisateur sous la forme d'un nombre à virgule flottante . Les agents utilisateurs ne doivent pas autoriser l'utilisateur à définir la valeur sur la chaîne vide.

**Validation des contraintes** : alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en un nombre à virgule flottante valide , le contrôle souffre d'une mauvaise entrée .

L' value attribut, s'il est spécifié, doit avoir une valeur qui est un nombre à virgule flottante valide .

L' algorithme de nettoyage de valeur est le suivant : Si la valeur de l'élément n'est pas un nombre à virgule flottante valide , définissez-le sur la meilleure représentation, en tant que nombre à virgule flottante , de la valeur par défaut .

La **valeur par défaut** est le minimum plus la moitié de la différence entre le minimum et le maximum , sauf si le maximum est inférieur au minimum , auquel cas la valeur par défaut est le minimum .

Lorsque l'élément souffre d'un sous-dépassement , l'agent utilisateur doit définir la valeur de l'élément sur la meilleure représentation, sous forme de nombre à virgule flottante , du minimum .

Lorsque l'élément [souffre d'un débordement](#) , si le [maximum](#) n'est pas inférieur au [minimum](#) , l'agent utilisateur doit définir la [valeur](#) de l'élément sur un [nombre à virgule flottante valide](#) qui représente le [maximum](#) .

Lorsque l'élément [souffre d'une discordance d'étape](#) , l'agent utilisateur doit arrondir la [valeur](#) de l'élément au nombre le plus proche pour lequel l'élément ne [souffrirait pas d'une discordance d'étape](#) , et qui est supérieur ou égal au [minimum](#) , et, si le [maximum](#) n'est pas inférieur au [minimum](#) , qui est inférieur ou égal au [maximum](#) , s'il existe un nombre qui correspond à ces contraintes. Si deux nombres correspondent à ces contraintes, alors les agents utilisateurs doivent utiliser celui qui est le plus proche de l'infini positif.

Par exemple, le balisage `<input type="range" min=0 max=100 step=20 value=50>` donne un contrôle de plage dont la valeur initiale est 60.

Voici un exemple de contrôle de plage utilisant une liste de saisie semi-automatique avec l' [list](#) attribut. Cela peut être utile s'il existe des valeurs sur toute la plage de la commande qui sont particulièrement importantes, telles que des niveaux d'éclairage préconfigurés ou des limites de vitesse typiques dans une commande de plage utilisée comme commande de vitesse. Le fragment de balisage suivant :

```
<input type="range" min="-100" max="100" value="0" step="10"
name="power" list="powers">
<datalist id="powers">
  <option value="0">
  <option value="-30">
  <option value="30">
  <option value="++50">
</datalist>
```

...avec la feuille de style suivante appliquée :

```
input { height: 75px; width: 49px; background: #D5CCBB; color:
black; }
```

... pourrait s'afficher comme :



Notez comment l'UA a déterminé l'orientation du contrôle à partir du rapport entre les propriétés de hauteur et de largeur spécifiées par la feuille de style. Les couleurs étaient également dérivées de la feuille de style. Les graduations, cependant, ont été dérivées du balisage. En particulier, l' [step](#) attribut n'a pas affecté le placement des

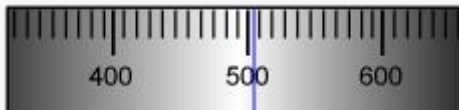
graduations, l'UA décidant de n'utiliser que les valeurs d'achèvement spécifiées par l'auteur, puis d'ajouter des graduations plus longues aux extrêmes.

Notez également comment la valeur invalide ++50a été ignorée.

Pour un autre exemple, considérez le fragment de balisage suivant :

```
<input name=x type=range min=100 max=700 step=9.09090909  
value=509.090909>
```

Un agent utilisateur peut s'afficher de différentes manières, par exemple :



Ou, alternativement, par exemple :

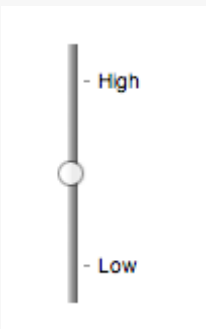


L'agent utilisateur peut choisir lequel afficher en fonction des dimensions données dans la feuille de style. Cela lui permettrait de conserver la même résolution pour les graduations, malgré les différences de largeur.

Enfin, voici un exemple de contrôle de plage avec deux valeurs étiquetées :

```
<input type="range" name="a" list="a-values">  
<datalist id="a-values">  
<option value="10" label="Low">  
<option value="90" label="High">  
</datalist>
```

Avec des styles qui font que le contrôle est dessiné verticalement, il peut ressembler à ceci :



*Dans cet état, les contraintes de plage et de pas sont appliquées même pendant la saisie de l'utilisateur, et il n'y a aucun moyen de définir la valeur sur la chaîne vide.*

L' minattribut, s'il est spécifié, doit avoir une valeur qui est un nombre à virgule flottante valide . La valeur minimale par défaut est 0. L' maxattribut, s'il est spécifié, doit avoir une valeur qui est un nombre à virgule flottante valide . Le maximum par défaut est 100.

Le facteur d'échelle de pas est 1. Le pas par défaut est 1 (autorisant uniquement les entiers, sauf si l' minattribut a une valeur non entière).

L' algorithme pour convertir une chaîne en nombre , étant donné une entrée de chaîne , est le suivant : Si l'application des règles d'analyse des valeurs de nombres à virgule flottante aux résultats d' entrée génère une erreur, alors renvoie une erreur ; sinon, renvoie le nombre résultant.

L' algorithme pour convertir un nombre en chaîne , étant donné un nombre en entrée , est le suivant : Renvoie la meilleure représentation , sous forme de nombre à virgule flottante , de input .

Les attributs de contenu d'élément communs input, les attributs IDL et les méthodes suivants s'appliquent à l'élément : autocomplete, list, max, min et step les attributs de contenu ; list, value et valueAsNumber attributs IDL ; stepDown() et stepUp() méthodes.

L' valueattribut IDL est en mode value .

Les événements input et s'appliquent .change

Les attributs de contenu suivants ne doivent pas être spécifiés et ne s'appliquent pas à l'élément : accept, alt, checked, dirname, formaction, formenctype, formmethod, formnovalidate, formtarget, height, maxlength, minlength, multiple, pattern, placeholder, popoverhidetarget, popovershowtarget, popovertoggletarget, readonly, required, size, src et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément : attributs checked, files, , , et IDL selectionStart; , , et méthodes selectionEnd selectionDirection valueAsDate select() setRangeText() setSelectionRange()

#### 4.10.5.1.14 État de la couleur `type=color()`



Lorsque l'attribut input d'un élément type est dans l' état Couleur , les règles de cette section s'appliquent.

L' inputélément représente un contrôle de puits de couleur, pour définir la valeur de l'élément sur une chaîne représentant une couleur simple .

*Dans cet état, il y a toujours une couleur sélectionnée et il n'y a aucun moyen de définir la valeur sur la chaîne vide.*

Si l'élément est mutable, l'agent utilisateur doit permettre à l'utilisateur de changer la couleur représentée par sa valeur, telle qu'obtenue en appliquant les règles d'analyse des valeurs de couleur simples. Les agents utilisateurs ne doivent pas permettre à l'utilisateur de définir la valeur sur une chaîne qui n'est pas une couleur simple en minuscule valide. Si l'agent utilisateur fournit une interface utilisateur pour sélectionner une couleur, alors la valeur doit être définie sur le résultat de l'utilisation des règles de sérialisation des valeurs de couleur simples selon la sélection de l'utilisateur. Les agents utilisateurs ne doivent pas autoriser l'utilisateur à définir la valeur sur la chaîne vide.

Le comportement d'activation d'entrée pour un tel élément *element* consiste à afficher le sélecteur, le cas échéant, pour *element*.

**Validation de contrainte** : alors que l'interface utilisateur décrit une entrée que l'agent utilisateur ne peut pas convertir en une couleur simple en minuscule valide, le contrôle souffre d'une mauvaise entrée.

L' valueattribut, s'il est spécifié et non vide, doit avoir une valeur qui est une couleur simple valide.

L' algorithme de nettoyage de valeur est le suivant : Si la valeur de l'élément est une couleur simple valide, alors réglez-la sur la valeur de l'élément convertie en minuscules ASCII ; sinon, réglez-le sur la chaîne "#000000".

input Les attributs de contenu d'élément et les attributs IDL communs suivants s'appliquent à l'élément : autocomplete et list attributs de contenu ; list et value les attributs IDL ; select() méthode.

L' valueattribut IDL est en mode value.

Les événements input et s'appliquent change

Les attributs de contenu suivants ne doivent pas être spécifiés et ne s'appliquent pas à l'élément : accept, alt, checked, dirname, formaction, formenctype, formmethod, formnovalidate, formtarget, height, max, maxlength, min, minlength, multiple, pattern, placeholder, popoverhidetarget, popovershowtarget, popovertoggletarget, readonly, required, size, src, step et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément : checked, files, selectionStart, selectionEnd, selectionDirection, valueAsDate et valueAsNumber attributs IDL ; setRangeText(), setSelectionRange(), stepDown() et stepUp() méthodes.

#### 4.10.5.1.15 État de la case à cocher type=checkbox()

Lorsque l'attribut `input` d'un élément `type` est à l'état [Case à cocher](#), les règles de cette section s'appliquent.

L'élément `input` [représente](#) un contrôle à deux états qui représente l'état [de vérification](#) de l'élément. Si l'état [de vérification](#) de l'élément est vrai, le contrôle représente une sélection positive, et s'il est faux, une sélection négative. Si l'attribut IDL `indeterminate` de l'élément est défini sur `true`, la sélection du contrôle doit être masquée comme si le contrôle était dans un troisième état, indéterminé.

*Le contrôle n'est jamais un véritable contrôle à trois états, même si l'attribut IDL `indeterminate` de l'élément est défini sur `true`. L'attribut IDL `indeterminate` ne donne que l'apparence d'un troisième état.*

Le [comportement d'activation des entrées](#) consiste à exécuter les étapes suivantes :

1. Si l'élément n'est pas [connecté](#), alors retournez.
2. [Lancez un événement](#) nommé `input` au niveau de l'élément avec les attributs `bubbles` et `composed` initialisés à `true`.
3. [Lancez un événement](#) nommé `change` au niveau de l'élément avec l'attribut `bubbles` initialisé à `true`.

**Validation de contrainte** : si l'élément est [requis](#) et que sa [vérification](#) est fausse, alors l'élément [souffre d'être manquant](#).

`input.indeterminate [ = value ]`

Lorsqu'il est défini, remplace le rendu des contrôles [de case à cocher](#) afin que la valeur actuelle ne soit pas visible.

L'élément `input` Les attributs de contenu d'élément communs et les attributs IDL suivants [s'appliquent](#) à l'élément : `checked`, et `required` les attributs de contenu ; `checked` et `value` les attributs IDL.

L'attribut IDL `value` est en mode [default/on](#).

Les événements `input` et [s'appliquent](#) `change`

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : `accept`, `alt`, `autocomplete`, `dirname`, `formaction`, `formenctype`, `formmethod`, `formnovalidate`, `formtarget`, `height`, `list`, `max`, `maxlength`, `min`, `minlength`, `multiple`, `pattern`, `placeholder`, `popoverhidetarget`, `popovershowtarget`, `popovertoggletarget`, `readonly`, `size`, `src`, `step` et `width`.

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément : `files`, `list`, `selectionStart`, `selectionEnd`, `selectionDirection`, `valueAsDate` et `valueAsNumber` IDL ; `select()`, `setRangeText()`, et méthodes `setSelectionRange()`, `stepDown()` et `stepUp()`

#### 4.10.5.1.16 État du bouton radio `type=radio` ( )



Lorsque l'attribut `input` d'un élément `type` est dans l'état [Bouton radio](#), les règles de cette section s'appliquent.

L'élément `input` représente un contrôle qui, lorsqu'il est utilisé conjointement avec d'autres éléments `input`, forme un [groupe de boutons radio](#) dans lequel un seul contrôle peut avoir son état [de vérification](#) défini sur vrai. Si l'état [de vérification](#) de l'élément est vrai, le contrôle représente le contrôle sélectionné dans le groupe, et s'il est faux, il indique un contrôle dans le groupe qui n'est pas sélectionné.

Le **groupe de boutons radio** qui contient un élément `input` a également tous les autres éléments `input` *b* qui remplissent toutes les conditions suivantes :

- L'attribut de l'élément `input` `type` est dans l'état [Bouton radio](#).
- Soit *a* et *b* ont le même [propriétaire de formulaire](#), soit ils n'ont pas [de propriétaire de formulaire](#).
- *a* et *b* sont dans le même [arbre](#).
- Ils ont tous les deux un `name` attribut, leurs `name` attributs ne sont pas vides et la valeur de l'attribut *a* `name` est égale à la valeur de l'attribut *b* `name`.

Un [arbre](#) ne doit pas contenir un élément `input` dont [le groupe de boutons radio](#) ne contient que cet élément.

Lorsque l'un des phénomènes suivants se produit, si l'état [de vérification](#) de l'élément est vrai après l'occurrence, l'état [de vérification](#) de tous les autres éléments du même [groupe de boutons radio](#) doit être défini sur faux :

- L'état [de vérification](#) de l'élément est défini sur true (quelle qu'en soit la raison).
- L'attribut `name` de l'élément est défini, modifié ou supprimé.
- Le [propriétaire du formulaire](#) de l'élément change.
- [Un changement de type est signalé](#) pour l'élément.

Le [comportement d'activation des entrées](#) consiste à exécuter les étapes suivantes :

1. Si l'élément n'est pas [connecté](#), alors retournez.
2. [Lancez un événement](#) nommé `input` au niveau de l'élément avec les attributs `bubbles` et `composed` initialisés à true.

3. Lancez un événement nommé change au niveau de l'élément avec l'attribut bubbles initialisé à true.

**Validation de contrainte** : si un élément du groupe de boutons radio est requis et que tous les input éléments du groupe de boutons radio ont une vérification qui est fausse, alors l'élément souffre d'être manquant .

L'exemple suivant, pour une raison quelconque, a spécifié que les puppeurs sont à la fois obligatoires et désactivés :

```
<form>

  <p><label><input type="radio" name="dog-type" value="pupper"
required disabled> Pupper</label>

  <p><label><input type="radio" name="dog-type" value="doggo">
Doggo</label>

  <p><button>Make your choice</button>

</form>
```

Si l'utilisateur essaie de soumettre ce formulaire sans d'abord sélectionner "Doggo", alors les deux input éléments souffriront d'être manquants , car un élément du groupe de boutons radio est requis (c'est-à-dire le premier élément), et les deux éléments du groupe de boutons radio ont une fausse vérification .

D'un autre côté, si l'utilisateur sélectionne "Doggo" puis soumet le formulaire, aucun input élément ne souffrira d'être manquant , car si l'un d'eux est requis , tous n'ont pas une fausse vérification .

*Si aucun des boutons radio d'un groupe de boutons radio n'est coché, ils seront tous initialement décochés dans l'interface, jusqu'à ce que l'un d'entre eux soit coché (soit par l'utilisateur, soit par le script).*

input Les attributs de contenu d'élément et les attributs IDL communs suivants s'appliquent à l'élément : checked et required attributs de contenu ; checked et value les attributs IDL.

L'attribut IDL value est en mode default/on .

Les événements input et s'appliquent change

Les attributs de contenu suivants ne doivent pas être spécifiés et ne s'appliquent pas à l'élément : accept, alt, autocomplete, dirname, formaction, formenctype, formmethod, formnovalidate, formtarget, height, list, max, maxlength, min, minlength, multiple, pattern, placeholder, popoverhidetarget, popovershowtarget, popovertoggletarget, readonly, size, src, step et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément : files, list, selectionStart, selectionEnd, selectionDirection, valueAsDate et valueAsNumber IDL ; select(), setRangeText(), et méthodes setSelectionRange(), stepDown() stepUp()



#### 4.10.5.1.17 État de téléchargement de fichier<sub>type=file</sub> ( )



Lorsque l'attribut input d'un élément type est à l'état [Téléchargement de fichier](#), les règles de cette section s'appliquent.

L' input élément [représente](#) une liste de **fichiers sélectionnés**, chaque fichier étant composé d'un nom de fichier, d'un type de fichier et d'un corps de fichier (le contenu du fichier).

Les noms de fichiers ne doivent pas contenir [de composants de chemin](#), même dans le cas où un utilisateur a sélectionné une hiérarchie de répertoires entière ou plusieurs fichiers portant le même nom dans différents répertoires. **Les composants de chemin**, pour les besoins de l'état [Téléchargement de fichier](#), sont les parties des noms de fichiers qui sont séparées par des caractères U+005C REVERSE SOLIDUS (\).

À moins que l' multiple attribut ne soit défini, il ne doit pas y avoir plus d'un fichier dans la liste des [fichiers sélectionnés](#).

Le [comportement d'activation d'entrée](#) pour un tel élément *element* consiste à [afficher le sélecteur, le cas échéant](#), pour *element*.

Si l'élément est [mutable](#), l'agent utilisateur doit également permettre à l'utilisateur de modifier les fichiers de la liste d'autres manières, par exemple en ajoutant ou en supprimant des fichiers par glisser-déposer. Lorsque l'utilisateur le fait, l'agent utilisateur doit [mettre à jour la sélection de fichiers](#) pour l'élément.

Si l'élément n'est pas [mutable](#), l'agent utilisateur ne doit pas autoriser l'utilisateur à modifier la sélection de l'élément.

Pour **mettre à jour la sélection de fichiers** pour un *élément element* :

1. [Mettez en file d'attente une tâche d'élément](#) sur l' *élément* [source de tâche d'interaction utilisateur](#) donné et suivez les étapes suivantes :
  1. Mettre à jour [les fichiers sélectionnés](#) de l'*élément* afin qu'ils représentent la sélection de l'utilisateur.
  2. [Lancez un événement](#) nommé input au niveau de l' input élément, avec les attributs bubbles et composed initialisés à true.
  3. [Lancez un événement](#) nommé change au niveau de l' input élément, avec l' bubbles attribut initialisé à true.

**Validation des contraintes** : Si l'élément est [obligatoire](#) et que la liste des [fichiers sélectionnés](#) est vide, alors l'élément [souffre d'être manquant](#).



L' **accept** attribut peut être spécifié pour fournir aux agents utilisateurs une indication des types de fichiers qui seront acceptés.

S'il est spécifié, l'attribut doit consister en un [ensemble de jetons séparés par des virgules](#) , chacun d'entre eux devant être une correspondance [ASCII non sensible à la casse](#) pour l'un des éléments suivants :

**La chaîne " `audio/*` "**

Indique que les fichiers audio sont acceptés.

**La chaîne " `video/*` "**

Indique que les fichiers vidéo sont acceptés.

**La chaîne " `image/*` "**

Indique que les fichiers image sont acceptés.

**Une [chaîne de type MIME valide sans paramètres](#)**

Indique que les fichiers du type spécifié sont acceptés.

**Une chaîne dont le premier caractère est un caractère U+002E FULL STOP (.)**

Indique que les fichiers avec l'extension de fichier spécifiée sont acceptés.

Les jetons ne doivent pas être des correspondances [ASCII insensibles à la casse](#) pour aucun des autres jetons (c'est-à-dire que les doublons ne sont pas autorisés). Pour obtenir la liste des jetons à partir de l'attribut, l'agent utilisateur doit [diviser la valeur de l'attribut par des virgules](#) .

Les agents utilisateurs peuvent utiliser la valeur de cet attribut pour afficher une interface utilisateur plus appropriée qu'un sélecteur de fichier générique. Par exemple, étant donné la valeur `image/*`, un agent utilisateur pourrait offrir à l'utilisateur la possibilité d'utiliser un appareil photo local ou de sélectionner une photographie dans sa collection de photos ; étant donné la valeur `audio/*`, un agent utilisateur pourrait offrir à l'utilisateur la possibilité d'enregistrer un clip à l'aide d'un microphone de casque.

Les agents utilisateurs doivent empêcher l'utilisateur de sélectionner des fichiers qui ne sont pas acceptés par un (ou plusieurs) de ces jetons.

*Les auteurs sont encouragés à spécifier à la fois les types MIME et les extensions correspondantes lorsqu'ils recherchent des données dans un format spécifique.*

Par exemple, considérez une application qui convertit des documents Microsoft Word en fichiers Open Document Format. Étant donné que les documents Microsoft Word

sont décrits avec une grande variété de types et d'extensions MIME, le site peut en répertorier plusieurs, comme suit :

```
<input type="file"
accept=".doc,.docx,.xml,application/msword,application/vnd.openxmlf
ormats-officedocument.wordprocessingml.document">
```

Sur les plates-formes qui n'utilisent que des extensions de fichiers pour décrire les types de fichiers, les extensions répertoriées ici peuvent être utilisées pour filtrer les documents autorisés, tandis que les types MIME peuvent être utilisés avec la table d'enregistrement des types du système (mappage des types MIME aux extensions utilisées par le système), le cas échéant, pour déterminer les autres extensions à autoriser. De même, sur un système qui n'a pas de noms de fichiers ou d'extensions mais étiquette les documents avec des types MIME en interne, les types MIME peuvent être utilisés pour sélectionner les fichiers autorisés, tandis que les extensions peuvent être utilisées si le système dispose d'une table d'enregistrement d'extensions qui mappe les extensions connues. aux types MIME utilisés par le système.

***Les extensions ont tendance à être ambiguës (par exemple, il existe un nombre incalculable de formats qui utilisent l' ".dat" extension " ", et les utilisateurs peuvent généralement renommer assez facilement leurs fichiers pour qu'ils aient une ".doc" extension " " même s'il ne s'agit pas de documents Microsoft Word), et les types MIME ont tendance à n'est pas fiable (par exemple, de nombreux formats n'ont pas de types officiellement enregistrés, et de nombreux formats sont en pratique étiquetés à l'aide d'un certain nombre de types MIME différents). Il est rappelé aux auteurs que, comme d'habitude, les données reçues d'un client doivent être traitées avec prudence, car elles peuvent ne pas être dans un format attendu même si l'utilisateur n'est pas hostile et que l'agent utilisateur a pleinement obéi aux exigences de l'attribut accept.***

## MDN

Pour des raisons historiques, l' valueattribut IDL préfixe le nom de fichier avec la chaîne " C:\fakepath\ ". Certains agents utilisateurs hérités incluait en fait le chemin complet (ce qui était une faille de sécurité). En conséquence, l'obtention du nom de fichier à partir de l' valueattribut IDL d'une manière rétrocompatible n'est pas triviale. La fonction suivante extrait le nom de fichier d'une manière convenablement compatible :

```
function extractFilename(path) {
    if (path.substr(0, 12) == "C:\\fakepath\\")
        return path.substr(12); // modern browser
    var x;
    x = path.lastIndexOf('/');
    if (x >= 0) // Unix-based path
        return path.substr(x+1);
    x = path.lastIndexOf('\\');
    if (x >= 0) // Windows-based path
```

```

    return path.substr(x+1);

    return path; // just the filename
}

```

Cela peut être utilisé comme suit :

```

<p><input type=file name=image
onchange="updateFilename(this.value)"></p>

<p>The name of the file you picked is: <span
id="filename">(none)</span></p>

<script>

    function updateFilename(path) {
        var name = extractFilename(path);
        document.getElementById('filename').textContent = name;
    }

</script>

```

input Les attributs de contenu d'élément communs et les attributs IDL suivants [s'appliquent](#) à l'élément : [accept](#), [multiple](#) et [required](#) les attributs de contenu ; [files](#) et [value](#) les attributs IDL ; [select\(\)](#) méthode.

L' [value](#) attribut IDL est en mode [filename](#) .

Les événements [input](#) et [s'appliquent](#) [.change](#)

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [alt](#), [autocomplete](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [list](#), [max](#), [maxlength](#), [min](#), [minlength](#), [pattern](#), [popoverhidetarget](#), [popovershowtarget](#), [popovertoggletarget](#), [placeholder](#), [readonly](#), [size](#), [src](#), [step](#) et [width](#).

L' [value](#) attribut de l'élément doit être omis.

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément : [checked](#), [list](#), [selectionStart](#), [selectionEnd](#), [selectionDirect](#) [ion](#), [valueAsDate](#) et [valueAsNumber](#) IDL ; [setRangeText\(\)](#), [setSelectionRange\(\)](#), [stepDown\(\)](#) et [stepUp\(\)](#) méthodes.

#### 4.10.5.1.18 État du bouton Soumettre `type=submit()`

Lorsque l'attribut input d'un élément type est à l'état Bouton Soumettre, les règles de cette section s'appliquent.

L' input élément représente un bouton qui, lorsqu'il est activé, soumet le formulaire. Si l'élément a un value attribut, l'étiquette du bouton doit être la valeur de cet attribut ; sinon, il doit s'agir d'une chaîne définie par l'implémentation qui signifie "Soumettre" ou quelque chose du genre. L'élément est un bouton, plus précisément un bouton d'envoi.

*Étant donné que l'étiquette par défaut est définie par l'implémentation et que la largeur du bouton dépend généralement de l'étiquette du bouton, la largeur du bouton peut laisser échapper quelques bits d'informations d'empreintes digitales. Ces bits sont susceptibles d'être fortement corrélés à l'identité de l'agent utilisateur et aux paramètres régionaux de l'utilisateur.*

Le comportement d'activation des entrées de l'élément est le suivant :

1. Si l'élément n'a pas de propriétaire de formulaire, alors retournez.
2. Si le document de nœud de l'élément n'est pas complètement actif, alors revenez.
3. Soumettez le propriétaire du formulaire à partir de l'élément.

Les attributs formation, formenctype, et sont formmethod des attributs de soumission de formulaire. formnovalidate formtarget

*L' formnovalidate attribut peut être utilisé pour créer des boutons de soumission qui ne déclenchent pas la validation de la contrainte.*

input Les attributs de contenu d'élément communs et les attributs IDL suivants s'appliquent à l'élément

: formation, formenctype, formmethod, formnovalidate, formtarget, popoverhidetarget, povertarget et povertoggetarget les attributs de contenu ; value Attribut IDL.

L' value attribut IDL est en mode par défaut.

Les attributs de contenu suivants ne doivent pas être spécifiés et ne s'appliquent pas à l'élément : accept, alt, autocomplete, checked, dirname, height, list, max, maxlength, min, minlength, multiple, pattern, placeholder, readonly, required, size, src, step et width.

Les attributs et méthodes IDL suivants ne s'appliquent pas à l'élément

: checked, files, list, selectionStart, selectionEnd, selectionDirection, valueAsDate et valueAsNumber IDL

; select(), setRangeText(), ,

et méthodes setSelectionRange(). stepDown() stepUp()

Les événements input et ne s'appliquent pas change

#### 4.10.5.1.19 État du bouton d'image `type=image ( )`



Lorsque l'attribut `input` d'un élément `type` est dans l'état [Image Button](#), les règles de cette section s'appliquent.

L'élément `input` représente soit une image à partir de laquelle un utilisateur peut sélectionner une coordonnée et soumettre le formulaire, soit un bouton à partir duquel l'utilisateur peut soumettre le formulaire. L'élément est un [bouton](#), plus précisément un [bouton d'envoi](#).

*La coordonnée est envoyée au serveur [lors de la soumission du formulaire](#) en envoyant deux entrées pour l'élément, dérivées du nom du contrôle mais avec ".x" et ".y" ajoutés au nom avec les composants x et y de la coordonnée respectivement.*



L'image est donnée par l'attribut `src`. L'attribut `src` doit être présent et doit contenir une [URL non vide valide potentiellement entourée d'espaces](#) faisant référence à une ressource d'image non interactive, éventuellement animée, qui n'est ni paginée ni scriptée.

Lorsque l'un de ces événements se produit

- l'attribut `input` de l'élément `type` est d'abord défini sur l'état [du bouton d'image](#) (éventuellement lorsque l'élément est créé pour la première fois) et l'attribut `src` est présent
- l'attribut `input` de l'élément `type` est remodifié à l'état [Image Button](#), et l'attribut `src` est présent, et sa valeur a changé depuis la dernière fois que l'attribut `type` était dans l'état [Image Button](#)
- l'attribut `input` de l'élément `type` est dans l'état [Image Button](#) et l'attribut `src` est défini ou modifié

alors, à moins que l'agent utilisateur ne puisse pas prendre en charge les images, ou que sa prise en charge des images ait été désactivée, ou que l'agent utilisateur ne récupère les images qu'à la demande, ou que la valeur de l'attribut `src` soit une chaîne vide, l'agent utilisateur doit [analyser](#) la valeur de la valeur de l'attribut `src`, relative au [nœud document](#) de l'élément, et si cela réussit, alors :

1. Soit *request* une nouvelle requête dont l'URL est l' enregistrement d'URL résultant , le client est l' objet de paramètres pertinent du document de nœud de l'élément , la destination est " " , le type d'initiateur est " " , le mode d'identification est " " et dont use-URL-credentials drapeau est défini.`imageinputinclude`
2. Fetch *request* , avec *processResponseEndOfBody* défini sur l'étape suivante en réponse à la réponse :
  1. Si le téléchargement a réussi et que l'image est disponible , mettez en file d'attente une tâche d'élément sur la source de la tâche d'interaction utilisateur en fonction de l' input élément pour déclencher un événement nommé load sur l' input élément.
  2. Sinon, si le processus de récupération échoue sans réponse du serveur distant, ou se termine mais que l'image n'est pas une image valide ou prise en charge, placez une tâche d'élément en file d'attente sur la source de la tâche d'interaction utilisateur en fonction de l' input élément pour déclencher un événement nommé error sur l' input élément .

La récupération de l'image doit retarder l'événement de chargement du document de nœud de l'élément jusqu'à ce que la tâche qui est mise en file d'attente par la source de tâche réseau une fois que la ressource a été récupérée (définie ci-dessous) a été exécutée.

Si l'image a été obtenue avec succès, sans erreur de réseau, et que le type d'image est un type d'image pris en charge, et que l'image est une image valide de ce type, alors l'image est dite **disponible** . Si cela est vrai avant que l'image ne soit entièrement téléchargée, chaque tâche mise en file d'attente par la source de tâches réseau pendant la récupération de l'image doit mettre à jour la présentation de l'image de manière appropriée.

L'agent utilisateur doit appliquer les règles de reniflage d'image pour déterminer le type de l'image, les en-têtes Content-Type associés à l'image donnant le *type officiel* . Si ces règles ne sont pas appliquées, le type de l'image doit être le type donné par les en-têtes Content-Type associés à l'image .

Les agents utilisateurs ne doivent pas prendre en charge les ressources autres que les images avec l' input élément. Les agents utilisateurs ne doivent pas exécuter de code exécutable intégré dans la ressource image. Les agents utilisateurs ne doivent afficher que la première page d'une ressource multipage. Les agents utilisateurs ne doivent pas permettre à la ressource d'agir de manière interactive, mais doivent respecter toute animation dans la ressource.

---



L' `alt` attribut fournit l'étiquette textuelle du bouton pour les utilisateurs et les agents utilisateurs qui ne peuvent pas utiliser l'image. L' `alt` attribut doit être présent, et doit contenir une chaîne non vide donnant le libellé qui serait approprié pour un bouton équivalent si l'image n'était pas disponible.

L' `input` élément prend en charge [les attributs de dimension](#) .

---

Si l' `src` attribut est défini, que l'image est [disponible](#) et que l'agent utilisateur est configuré pour afficher cette image, alors l'élément [représente](#) une commande pour sélectionner une [coordonnée](#) à partir de l'image spécifiée par l' `src` attribut. Dans ce cas, si l'élément est [mutable](#) , l'agent utilisateur doit autoriser l'utilisateur à sélectionner cette [coordonnée](#) .

Sinon, l'élément [représente](#) un bouton d'envoi dont le libellé est donné par la valeur de l' `alt` attribut.

[Le comportement d'activation des entrées](#) de l'élément est le suivant :

1. Si l'élément n'a pas de [propriétaire de formulaire](#) , alors retournez.
2. Si le [document de nœud](#) de l'élément n'est pas [complètement actif](#) , alors revenez.
3. Si l'utilisateur a activé le contrôle tout en sélectionnant explicitement une coordonnée, alors définissez la [coordonnée sélectionnée de l'élément](#) de l'élément sur cette coordonnée.

*Ceci n'est possible que dans les conditions énoncées ci-dessus, lorsque l'élément [représente](#) une commande de sélection d'une telle coordonnée. Même dans ce cas, l'utilisateur peut activer le contrôle sans sélectionner explicitement une coordonnée.*

4. [Soumettez](#) le [propriétaire du formulaire](#) à partir de l'élément.

La **coordonnée sélectionnée** de l'élément consiste en une composante x et une composante y . Il est initialement (0, 0). Les coordonnées représentent la position par rapport au bord de l'image de l'élément, l'espace de coordonnées ayant la direction x positive vers la droite et la direction y positive vers le bas.

Le composant x doit être un [entier valide](#) représentant un nombre x dans la plage  $-(border_{left} + padding_{left}) \leq x \leq width + border_{right} + padding_{right}$  , où *width* est la



largeur rendue de l'image,  $border_{left}$  est la largeur de la bordure à gauche de l'image,  $padding_{left}$  est la largeur de la bordure à gauche de l'image,  $border_{right}$  est la largeur de la bordure à droite de l'image, et  $padding_{right}$  est la largeur du rembourrage à droite de l'image, avec toutes les dimensions données en [pixels CSS](#).

Le composant  $y$  doit être un [entier valide](#) représentant un nombre  $y$  dans la plage  $-(border_{top} + padding_{top}) \leq y \leq height + border_{bottom} + padding_{bottom}$ , où  $height$  est la hauteur rendue de l'image,  $border_{top}$  est la largeur de la bordure au-dessus de l'image,  $padding_{top}$  est la largeur du padding au-dessus de l'image,  $border_{bottom}$  est la largeur de la bordure sous l'image, et  $padding_{bottom}$  est la largeur du rembourrage sous l'image, avec toutes les dimensions données en [pixels CSS](#).

Lorsqu'une bordure ou un rembourrage est manquant, sa largeur est de zéro [pixel CSS](#).

---

Les attributs [formation](#), [formenctype](#), et sont [formmethod](#) des [attributs de soumission de formulaire](#). [formnovalidate](#) [formtarget](#)

```
image.width [ = value ]  
image.height [ = value ]
```

Ces attributs renvoient les dimensions réelles rendues de l'image, ou zéro si les dimensions ne sont pas connues.

Ils peuvent être définis pour modifier les attributs de contenu correspondants.

[input](#) Les attributs de contenu d'élément communs et les attributs IDL suivants [s'appliquent](#) à l'élément

: [alt](#), [formation](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [popoverhidetarget](#), [popovershowtarget](#), [popovetoggletarget](#), [src](#) et [width](#) les attributs de contenu ; [value](#) Attribut IDL.

L' [value](#) attribut IDL est en mode [par défaut](#).

Les attributs de contenu suivants ne doivent pas être spécifiés et [ne s'appliquent pas](#) à l'élément : [accept](#), [autocomplete](#), [checked](#), [dirname](#), [list](#), [max](#), [maxlength](#), [min](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [readonly](#), [required](#), [size](#) et [step](#).

L' [value](#) attribut de l'élément doit être omis.

Les attributs et méthodes IDL suivants [ne s'appliquent pas](#) à l'élément

: [checked](#), [files](#), [list](#), [selectionStart](#), [selectionEnd](#), [selectionDirection](#), [valueAsDate](#) et [valueAsNumber](#) IDL ; [select\(\)](#), [setRangeText\(\)](#), ,

et méthodes [setSelectionRange\(\)](#). [stepDown\(\)](#) [stepUp\(\)](#)

Les événements [input](#) et [ne s'appliquent pas](#) [.change](#)

De nombreux aspects du comportement de cet état sont similaires au comportement de l'[img](#) élément. Les lecteurs sont encouragés à lire cette section, où bon nombre des mêmes exigences sont décrites plus en détail.

Prenez le formulaire suivant :

```
<form action="process.cgi">
  <input type=image src=map.png name=where alt="Show location list">
</form>
```

Si l'utilisateur a cliqué sur l'image à la coordonnée (127,40), l'URL utilisée pour soumettre le formulaire serait " `process.cgi?where.x=127&where.y=40`".

(In this example, it's assumed that for users who don't see the map, and who instead just see a button labeled "Show location list", clicking the button will cause the server to show a list of locations to pick from instead of the map.)

#### 4.10.5.1.20 Reset Button state (`type=reset`)



When an [input](#) element's [type](#) attribute is in the [Reset Button](#) state, the rules in this section apply.

The [input](#) element [represents](#) a button that, when activated, resets the form. If the element has a [value](#) attribute, the button's label must be the value of that attribute; otherwise, it must be an [implementation-defined](#) string that means "Reset" or some such. The element is a [button](#).

*Since the default label is [implementation-defined](#), and the width of the button typically depends on the button's label, the button's width can leak a few bits of fingerprintable information. These bits are likely to be strongly correlated to the identity of the user agent and the user's locale.*

The element's [input activation behavior](#) is as follows:

1. If the element does not have a [form owner](#), then return.
2. If the element's [node document](#) is not [fully active](#), then return.
3. [Reset](#) the [form owner](#) from the element.

**Constraint validation:** The element is [barred from constraint validation](#).

The [value](#) IDL attribute [applies](#) to this element and is in mode [default](#).

The following common [input](#) element content attributes [apply](#) to the element: [popoverhidetarget](#), [popovershowtarget](#), and [popovertoggletarget](#).

The following content attributes must not be specified and [do not apply](#) to the element: [accept](#), [alt](#), [autocomplete](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [list](#), [max](#), [maxlength](#), [min](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [readonly](#), [required](#), [size](#), [src](#), [step](#), and [width](#).

The following IDL attributes and methods [do not apply](#) to the element: [checked](#), [files](#), [list](#), [selectionStart](#), [selectionEnd](#), [selectionDirection](#), [valueAsDate](#), and [valueAsNumber](#) IDL attributes; [select\(\)](#), [setRangeText\(\)](#), [setSelectionRange\(\)](#), [stepDown\(\)](#), and [stepUp\(\)](#) methods.

The [input](#) and [change](#) events [do not apply](#).

#### 4.10.5.1.21 Button state (`type=button`)



When an [input](#) element's [type](#) attribute is in the [Button](#) state, the rules in this section apply.

The [input](#) element [represents](#) a button with no default behavior. A label for the button must be provided in the [value](#) attribute, though it may be the empty string. If the element has a [value](#) attribute, the button's label must be the value of that attribute; otherwise, it must be the empty string. The element is a [button](#).

The element has no [input activation behavior](#).

**Constraint validation:** The element is [barred from constraint validation](#).

The [value](#) IDL attribute [applies](#) to this element and is in mode [default](#).

The following common [input](#) element content attributes [apply](#) to the element: [popoverhidetarget](#), [popovershowtarget](#), and [popovertoggletarget](#).

The following content attributes must not be specified and [do not apply](#) to the element: [accept](#), [alt](#), [autocomplete](#), [checked](#), [dirname](#), [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), [formtarget](#), [height](#), [list](#), [max](#), [maxlength](#), [min](#), [minlength](#), [multiple](#), [pattern](#), [placeholder](#), [readonly](#), [required](#), [size](#), [src](#), [step](#), and [width](#).

The following IDL attributes and methods [do not apply](#) to the element: [checked](#), [files](#), [list](#), [selectionStart](#), [selectionEnd](#), [selectionDirection](#), [valueAsDate](#), and [valueAsNumber](#) IDL

attributes; [select\(\)](#), [setRangeText\(\)](#), [setSelectionRange\(\)](#), [stepDown\(\)](#), and [stepUp\(\)](#) methods.

The [input](#) and [change](#) events [do not apply](#).

#### 4.10.5.2 Implementation notes regarding localization of form controls

*This section is non-normative.*

The formats shown to the user in date, time, and number controls is independent of the format used for form submission.

Browsers are encouraged to use user interfaces that present dates, times, and numbers according to the conventions of either the locale implied by the [input](#) element's [language](#) or the user's preferred locale. Using the page's locale will ensure consistency with page-provided data.

For example, it would be confusing to users if an American English page claimed that a Cirque De Soleil show was going to be showing on 02/03, but their browser, configured to use the British English locale, only showed the date 03/02 in the ticket purchase date picker. Using the page's locale would at least ensure that the date was presented in the same format everywhere. (There's still a risk that the user would end up arriving a month late, of course, but there's only so much that can be done about such cultural differences...)

#### 4.10.5.3 Common [input](#) element attributes

These attributes only [apply](#) to an [input](#) element if its [type](#) attribute is in a state whose definition declares that the attribute [applies](#). When an attribute [doesn't apply](#) to an [input](#) element, user agents must [ignore](#) the attribute, regardless of the requirements and definitions below.

##### 4.10.5.3.1 The [maxlength](#) and [minlength](#) attributes



The [maxlength](#) attribute, when it [applies](#), is a [form control](#) [maxlength attribute](#).



The **minlength** attribute, when it [applies](#), is a [form control minlength attribute](#).

If the **input** element has a [maximum allowed value length](#), then the [length](#) of the value of the element's **value** attribute must be equal to or less than the element's [maximum allowed value length](#).

The following extract shows how a messaging client's text entry could be arbitrarily restricted to a fixed number of characters, thus forcing any conversation through this medium to be terse and discouraging intelligent discourse.

```
<label>What are you doing? <input name=status  
maxlength=140></label>
```

Here, a password is given a minimum length:

```
<p><label>Username: <input name=u required></label>  
<p><label>Password: <input name=p required minlength=12></label>
```

#### 4.10.5.3.2 The **size** attribute

The **size** attribute gives the number of characters that, in a visual rendering, the user agent is to allow the user to see while editing the element's [value](#).

The **size** attribute, if specified, must have a value that is a [valid non-negative integer](#) greater than zero.

If the attribute is present, then its value must be parsed using the [rules for parsing non-negative integers](#), and if the result is a number greater than zero, then the user agent should ensure that at least that many characters are visible.

The **size** IDL attribute is [limited to only non-negative numbers greater than zero](#) and has a default value of 20.

#### 4.10.5.3.3 The **readonly** attribute



The **readonly** attribute is a [boolean attribute](#) that controls whether or not the user can edit the form control. When specified, the element is not [mutable](#).

**Constraint validation:** If the **readonly** attribute is specified on an **input** element, the element is [barred from constraint validation](#).

The difference between disabled and readonly is that read-only controls can still function, whereas disabled controls generally do not function as controls until they are enabled. This is spelled out in more detail elsewhere in this specification with normative requirements that refer to the disabled concept (for example, the element's activation behavior, whether or not it is a focusable area, or when constructing the entry list). Any other behavior related to user interaction with disabled controls, such as whether text can be selected or copied, is not defined in this standard.

Only text controls can be made read-only, since for other controls (such as checkboxes and buttons) there is no useful distinction between being read-only and being disabled, so the readonly attribute does not apply.

In the following example, the existing product identifiers cannot be modified, but they are still displayed as part of the form, for consistency with the row representing a new product (where the identifier is not yet filled in).

```
<form action="products.cgi" method="post" enctype="multipart/form-
data">
  <table>
    <tr> <th> Product ID <th> Product name <th> Price <th> Action
    <tr>
      <td> <input readonly="readonly" name="1.pid" value="H412">
      <td> <input required="required" name="1.pname" value="Floor lamp
Ulke">
      <td> $<input required="required" type="number" min="0"
step="0.01" name="1.pprice" value="49.99">
      <td> <button formnovalidate="formnovalidate" name="action"
value="delete:1">Delete</button>
    <tr>
      <td> <input readonly="readonly" name="2.pid" value="FG28">
      <td> <input required="required" name="2.pname" value="Table lamp
Ulke">
      <td> $<input required="required" type="number" min="0"
step="0.01" name="2.pprice" value="24.99">
      <td> <button formnovalidate="formnovalidate" name="action"
value="delete:2">Delete</button>
    <tr>
      <td> <input required="required" name="3.pid" value=""
pattern="[A-Z0-9]+">
      <td> <input required="required" name="3.pname" value="">
      <td> $<input required="required" type="number" min="0"
step="0.01" name="3.pprice" value="">
      <td> <button formnovalidate="formnovalidate" name="action"
value="delete:3">Delete</button>
    </table>
    <p> <button formnovalidate="formnovalidate" name="action"
value="add">Add</button> </p>
```

```
<p> <button name="action" value="update">Save</button> </p>
</form>
```

#### 4.10.5.3.4 The required attribute

The **required** attribute is a [boolean attribute](#). When specified, the element is **required**.

**Constraint validation:** If the element is [required](#), and its [value](#) IDL attribute [applies](#) and is in the mode [value](#), and the element is [mutable](#), and the element's [value](#) is the empty string, then the element is [suffering from being missing](#).

The following form has two required fields, one for an email address and one for a password. It also has a third field that is only considered valid if the user types the same password in the password field and this third field.

```
<h1>Create new account</h1>
<form action="/newaccount" method=post
  oninput="up2.setCustomValidity(up2.value != up.value ?
  'Passwords do not match.' : '')">
  <p>
    <label for="username">Email address:</label>
    <input id="username" type=email required name=un>
  <p>
    <label for="password1">Password:</label>
    <input id="password1" type=password required name=up>
  <p>
    <label for="password2">Confirm password:</label>
    <input id="password2" type=password name=up2>
  <p>
    <input type=submit value="Create account">
</form>
```

For radio buttons, the required attribute is satisfied if any of the radio buttons in the [group](#) is selected. Thus, in the following example, any of the radio buttons can be checked, not just the one marked as required:

```
<fieldset>
  <legend>Did the movie pass the Bechdel test?</legend>
  <p><label><input type="radio" name="bechdel" value="no-
  characters"> No, there are not even two female characters in the
  movie. </label>
  <p><label><input type="radio" name="bechdel" value="no-names"> No,
  the female characters never talk to each other. </label>
```

```

<p><label><input type="radio" name="bechdel" value="no-topic"> No,
when female characters talk to each other it's always about a male
character. </label>

<p><label><input type="radio" name="bechdel" value="yes" required>
Yes. </label>

<p><label><input type="radio" name="bechdel" value="unknown"> I
don't know. </label>

</fieldset>

```

To avoid confusion as to whether a [radio button group](#) is required or not, authors are encouraged to specify the attribute on all the radio buttons in a group. Indeed, in general, authors are encouraged to avoid having radio button groups that do not have any initially checked controls in the first place, as this is a state that the user cannot return to, and is therefore generally considered a poor user interface.

#### 4.10.5.3.5 The multiple attribute



The **multiple** attribute is a [boolean attribute](#) that indicates whether the user is to be allowed to specify more than one value.

The following extract shows how an email client's "To" field could accept multiple email addresses.

```

<label>To: <input type=email multiple name=to></label>

```

If the user had, amongst many friends in their user contacts database, two friends "Spider-Man" (with address "spider@parker.example.net") and "Scarlet Witch" (with address "scarlet@avengers.example.net"), then, after the user has typed "s", the user agent might suggest these two email addresses to the user.

The page could also link in the user's contacts database from the site:

```

<label>To: <input type=email multiple name=to
list=contacts></label>

...

<datalist id="contacts">
  <option value="hedral@damowmow.com">
  <option value="pillar@example.com">

```



```
<option value="astrophysics@example.net">
<option value="astronomy@science.example.org">
</datalist>
```

Suppose the user had entered "bob@example.net" into this text control, and then started typing a second email address starting with "s". The user agent might show both the two friends mentioned earlier, as well as the "astrophysics" and "astronomy" values given in the [datalist](#) element.

The following extract shows how an email client's "Attachments" field could accept multiple files for upload.

```
<label>Attachments: <input type=file multiple name=att></label>
```

#### 4.10.5.3.6 The [pattern](#) attribute



The [pattern](#) attribute specifies a regular expression against which the control's [value](#), or, when the [multiple](#) attribute [applies](#) and is set, the control's [values](#), are to be checked.

If specified, the attribute's value must match the JavaScript [Pattern](#)<sub>(+UnicodeMode, +N]</sub> production.

The **compiled pattern regular expression** of an [input](#) element, if it exists, is a JavaScript [RegExp](#) object. It is determined as follows:

1. If the element does not have a [pattern](#) attribute specified, then return nothing. The element has no [compiled pattern regular expression](#).
2. Let *pattern* be the value of the [pattern](#) attribute of the element.
3. Let *regexpCompletion* be [RegExpCreate](#)(*pattern*, "u"). [\[JAVASCRIPT\]](#)
4. If *regexpCompletion* is an [abrupt completion](#), then return nothing. The element has no [compiled pattern regular expression](#).

*User agents are encouraged to log this error in a developer console, to aid debugging.*

5. Let *anchoredPattern* be the string `"^(?:",` followed by *pattern*, followed by `)$"`.
6. Return ! [RegExpCreate](#)(*anchoredPattern*, "u").

*The reasoning behind these steps, instead of just using the value of the [pattern](#) attribute directly, is twofold. First, we want to ensure that when matched against a string, the regular expression's start is anchored to the start of the string and its end to the end of the string. Second, we want to ensure that the regular expression is valid in standalone form, instead of only becoming valid after being surrounded by the `"^(?:"` and `)$"` anchors.*

A [RegExp](#) object *regexp* **matches** a string *input*, if  
! [RegExpBuiltinExec](#)(*regexp*, *input*) is not null.

**Constraint validation:** If the element's [value](#) is not the empty string, and either the element's [multiple](#) attribute is not specified or it [does not apply](#) to the [input](#) element given its [type](#) attribute's current state, and the element has a [compiled pattern regular expression](#) but that regular expression does not [match](#) the element's [value](#), then the element is [suffering from a pattern mismatch](#).

**Constraint validation:** If the element's [value](#) is not the empty string, and the element's [multiple](#) attribute is specified and [applies](#) to the [input](#) element, and the element has a [compiled pattern regular expression](#) but that regular expression does not [match](#) each of the element's [values](#), then the element is [suffering from a pattern mismatch](#).

When an [input](#) element has a [pattern](#) attribute specified, authors should include a **title** attribute to give a description of the pattern. User agents may use the contents of this attribute, if it is present, when informing the user that the pattern is not matched, or at any other suitable time, such as in a tooltip or read out by assistive technology when the control [gains focus](#).

For example, the following snippet:

```
<label> Part number:
  <input pattern="[0-9][A-Z]{3}" name="part"
    title="A part number is a digit followed by three uppercase
    letters."/>
</label>
```

...could cause the UA to display an alert such as:

```
A part number is a digit followed by three uppercase letters.
You cannot submit this form when the field is incorrect.
```

When a control has a [pattern](#) attribute, the [title](#) attribute, if used, must describe the pattern. Additional information could also be included, so long as it assists the user in filling in the control. Otherwise, assistive technology would be impaired.

For instance, if the title attribute contained the caption of the control, assistive technology could end up saying something like `The text you have entered does not match the required pattern. Birthday, which is not useful.`

UAs may still show the [title](#) in non-error situations (for example, as a tooltip when hovering over the control), so authors should be careful not to word [titles](#) as if an error has necessarily occurred.

#### 4.10.5.3.7 The [min](#) and [max](#) attributes



Some form controls can have explicit constraints applied limiting the allowed range of values that the user can provide. Normally, such a range would be linear and continuous. A form control can **have a periodic domain**, however, in which case the form control's broadest possible range is finite, and authors can specify explicit ranges within it that span the boundaries.

Specifically, the broadest range of a [type=time](#) control is midnight to midnight (24 hours), and authors can set both continuous linear ranges (such as 9pm to 11pm) and discontinuous ranges spanning midnight (such as 11pm to 1am).

The [min](#) and [max](#) attributes indicate the allowed range of values for the element.

Their syntax is defined by the section that defines the [type](#) attribute's current state.

If the element has a [min](#) attribute, and the result of applying the [algorithm to convert a string to a number](#) to the value of the [min](#) attribute is a number, then that number is the element's **minimum**; otherwise, if the [type](#) attribute's current state defines a **default minimum**, then that is the [minimum](#); otherwise, the element has no [minimum](#).

The [min](#) attribute also defines the [step base](#).

If the element has a [max](#) attribute, and the result of applying the [algorithm to convert a string to a number](#) to the value of the [max](#) attribute is a number, then that number is the element's **maximum**; otherwise, if the [type](#) attribute's current state defines a **default maximum**, then that is the [maximum](#); otherwise, the element has no [maximum](#).

If the element does not [have a periodic domain](#), the [max](#) attribute's value (the [maximum](#)) must not be less than the [min](#) attribute's value (its [minimum](#)).

*If an element that does not [have a periodic domain](#) has a [maximum](#) that is less than its [minimum](#), then so long as the element has a [value](#), it will either be [suffering from an underflow](#) or [suffering from an overflow](#).*

An element **has a reversed range** if it [has a periodic domain](#) and its [maximum](#) is less than its [minimum](#).

An element **has range limitations** if it has a defined [minimum](#) or a defined [maximum](#).

**Constraint validation:** When the element has a [minimum](#) and does not [have a reversed range](#), and the result of applying the [algorithm to convert a string to a number](#) to the string given by the element's [value](#) is a number, and the number obtained from that algorithm is less than the [minimum](#), the element is [suffering from an underflow](#).

**Constraint validation:** When the element has a [maximum](#) and does not [have a reversed range](#), and the result of applying the [algorithm to convert a string to a number](#) to the string given by the element's [value](#) is a number, and the number obtained from that algorithm is more than the [maximum](#), the element is [suffering from an overflow](#).

**Constraint validation:** When an element [has a reversed range](#), and the result of applying the [algorithm to convert a string to a number](#) to the string given by the element's [value](#) is a number, and the number obtained from that algorithm is more than the [maximum](#) and less than the [minimum](#), the element is simultaneously [suffering from an underflow](#) and [suffering from an overflow](#).

The following date control limits input to dates that are before the 1980s:

```
<input name=bday type=date max="1979-12-31">
```

The following number control limits input to whole numbers greater than zero:

```
<input name=quantity required="" type="number" min="1" value="1">
```

The following time control limits input to those minutes that occur between 9pm and 6am, defaulting to midnight:

```
<input name="sleepStart" type=time min="21:00" max="06:00"
step="60" value="00:00">
```

#### 4.10.5.3.8 The [step](#) attribute



The **step** attribute indicates the granularity that is expected (and required) of the **value** or **values**, by limiting the allowed values. The section that defines the **type** attribute's current state also defines the **default step**, the **step scale factor**, and in some cases the **default step base**, which are used in processing the attribute as described below.

The **step** attribute, if specified, must either have a value that is a [valid floating-point number](#) that [parses](#) to a number that is greater than zero, or must have a value that is an [ASCII case-insensitive](#) match for the string "any".

The attribute provides the **allowed value step** for the element, as follows:

1. If the attribute does not [apply](#), then there is no [allowed value step](#).
2. Otherwise, if the attribute is absent, then the [allowed value step](#) is the [default step](#) multiplied by the [step scale factor](#).
3. Otherwise, if the attribute's value is an [ASCII case-insensitive](#) match for the string "any", then there is no [allowed value step](#).
4. Otherwise, if the [rules for parsing floating-point number values](#), when they are applied to the attribute's value, return an error, zero, or a number less than zero, then the [allowed value step](#) is the [default step](#) multiplied by the [step scale factor](#).
5. Otherwise, the [allowed value step](#) is the number returned by the [rules for parsing floating-point number values](#) when they are applied to the attribute's value, multiplied by the [step scale factor](#).

The **step base** is the value returned by the following algorithm:

1. If the element has a **min** content attribute, and the result of applying the [algorithm to convert a string to a number](#) to the value of the **min** content attribute is not an error, then return that result.
2. If the element has a **value** content attribute, and the result of applying the [algorithm to convert a string to a number](#) to the value of the **value** content attribute is not an error, then return that result.
3. If a [default step base](#) is defined for this element given its **type** attribute's state, then return it.
4. Return zero.

**Constraint validation:** When the element has an [allowed value step](#), and the result of applying the [algorithm to convert a string to a number](#) to the string given by the element's **value** is a number, and that number subtracted from the [step base](#) is not an integral multiple of the [allowed value step](#), the element is [suffering from a step mismatch](#).

The following range control only accepts values in the range 0..1, and allows 256 steps in that range:

```
<input name=opacity type=range min=0 max=1 step=0.00392156863>
```

The following control allows any time in the day to be selected, with any accuracy (e.g. thousandth-of-a-second accuracy or more):

```
<input name=favtime type=time step=any>
```

Normally, time controls are limited to an accuracy of one minute.

#### 4.10.5.3.9 The [list](#) attribute



The **list** attribute is used to identify an element that lists predefined options suggested to the user.

If present, its value must be the [ID](#) of a [datalist](#) element in the same [tree](#).

The **suggestions source element** is the first element in the [tree](#) in [tree order](#) to have an [ID](#) equal to the value of the [list](#) attribute, if that element is a [datalist](#) element. If there is no [list](#) attribute, or if there is no element with that [ID](#), or if the first element with that [ID](#) is not a [datalist](#) element, then there is no [suggestions source element](#).

If there is a [suggestions source element](#), then, when the user agent is allowing the user to edit the [input](#) element's [value](#), the user agent should offer the suggestions represented by the [suggestions source element](#) to the user in a manner suitable for the type of control used. If appropriate, the user agent should use the suggestion's [label](#) and [value](#) to identify the suggestion to the user.

User agents are encouraged to filter the suggestions represented by the [suggestions source element](#) when the number of suggestions is large, including only the most relevant ones (e.g. based on the user's input so far). No precise threshold is defined, but capping the list at four to seven values is reasonable. If filtering based on the user's input, user agents should search within both the [label](#) and [value](#) of the suggestions for matches. User agents should consider how input variations affect the matching process. Unicode normalization should be applied so that different underlying Unicode code point sequences, caused by different keyboard- or input-specific mechanisms, do not interfere with the matching process. Case variations should be ignored, which may require language-specific case mapping. For examples of these, see *Character Model for the World Wide Web: String Matching*. User agents may also provide other matching features: for illustration, a few examples include matching different forms of kana to each other (or to kanji), ignoring accents, or applying spelling correction. [\[CHARMODNORM\]](#)

This text field allows you to choose a type of JavaScript function.

```
<input type="text" list="function-types">
<datalist id="function-types">
  <option value="function">function</option>
  <option value="async function">async function</option>
  <option value="function*">generator function</option>
  <option value="=>">arrow function</option>
  <option value="async =>">async arrow function</option>
  <option value="async function*">async generator function</option>
</datalist>
```

For user agents that follow the above suggestions, both the [label](#) and [value](#) would be shown:

Then, typing "arrow" or "=>" would filter the list to the entries with labels "arrow function" and "async arrow function". Typing "generator" or "\*" would filter the list to the entries with labels "generator function" and "async generator function".

*As always, user agents are free to make user interface decisions which are appropriate for their particular requirements and for the user's particular circumstances. However, this has historically been an area of confusion for implementers, web developers, and users alike, so we've given some "should" suggestions above.*

How user selections of suggestions are handled depends on whether the element is a control accepting a single value only, or whether it accepts multiple values:

**If the element does not have a [multiple](#) attribute specified or if the [multiple](#) attribute [does not apply](#)**

When the user selects a suggestion, the [input](#) element's [value](#) must be set to the selected suggestion's [value](#), as if the user had written that value themselves.

**If the element's [type](#) attribute is in the [Email](#) state and the element has a [multiple](#) attribute specified**

When the user selects a suggestion, the user agent must either add a new entry to the [input](#) element's [values](#), whose value is the selected suggestion's [value](#), or change an existing entry in the [input](#) element's [values](#) to have the value given by the selected suggestion's [value](#), as if the user had themselves added an entry with that value, or edited an existing entry to be that value. Which behavior is to be applied depends on the user interface in an [implementation-defined](#) manner.

---

If the [list](#) attribute [does not apply](#), there is no [suggestions source element](#).

This URL field offers some suggestions.

```
<label>Homepage: <input name=hp type=url list=hpurls></label>
<datalist id=hpurls>
  <option value="https://www.google.com/" label="Google">
  <option value="https://www.reddit.com/" label="Reddit">
</datalist>
```

Other URLs from the user's history might show also; this is up to the user agent.

This example demonstrates how to design a form that uses the autocompletion list feature while still degrading usefully in legacy user agents.

If the autocompletion list is merely an aid, and is not important to the content, then simply using a [datalist](#) element with children [option](#) elements is enough. To prevent the values from being rendered in legacy user agents, they need to be placed inside the [value](#) attribute instead of inline.

```
<p>
  <label>
    Enter a breed:
    <input type="text" name="breed" list="breeds">
    <datalist id="breeds">
      <option value="Abyssinian">
      <option value="Alpaca">
      <!-- ... -->
    </datalist>
  </label>
</p>
```



However, if the values need to be shown in legacy UAs, then fallback content can be placed inside the [datalist](#) element, as follows:

```
<p>
  <label>
    Enter a breed:
    <input type="text" name="breed" list="breeds">
  </label>
  <datalist id="breeds">
    <label>
      or select one from the list:
      <select name="breed">
        <option value=""> (none selected)
        <option>Abyssinian
        <option>Alpaca
        <!-- ... -->
      </select>
    </label>
  </datalist>
</p>
```

The fallback content will only be shown in UAs that don't support [datalist](#). The options, on the other hand, will be detected by all UAs, even though they are not children of the [datalist](#) element.

Note that if an [option](#) element used in a [datalist](#) is [selected](#), it will be selected by default by legacy UAs (because it affects the [select](#) element), but it will not have any effect on the [input](#) element in UAs that support [datalist](#).

#### 4.10.5.3.10 The [placeholder](#) attribute



The [placeholder](#) attribute represents a *short* hint (a word or short phrase) intended to aid the user with data entry when the control has no value. A hint could be a sample value or a brief description of the expected format. The attribute, if specified, must have a value that contains no U+000A LINE FEED (LF) or U+000D CARRIAGE RETURN (CR) characters.

The [placeholder](#) attribute should not be used as an alternative to a [label](#). For a longer hint or other advisory text, the [title](#) attribute is more appropriate.

*These mechanisms are very similar but subtly different: the hint given by the control's [label](#) is shown at all times; the short hint given in the [placeholder](#) attribute is shown before the user enters a value; and the hint in the [title](#) attribute is shown when the user requests further help.*

User agents should present this hint to the user, after having [stripped newlines](#) from it, when the element's [value](#) is the empty string, especially if the control is not [focused](#).

If a user agent normally doesn't show this hint to the user when the control is [focused](#), then the user agent should nonetheless show the hint for the control if it was focused as a result of the [autofocus](#) attribute, since in that case the user will not have had an opportunity to examine the control before focusing it.

Here is an example of a mail configuration user interface that uses the [placeholder](#) attribute:

```
<fieldset>
  <legend>Mail Account</legend>
  <p><label>Name: <input type="text" name="fullname"
placeholder="John Ratzenberger"></label></p>
  <p><label>Address: <input type="email" name="address"
placeholder="john@example.net"></label></p>
  <p><label>Password: <input type="password"
name="password"></label></p>
  <p><label>Description: <input type="text" name="desc"
placeholder="My Email Account"></label></p>
</fieldset>
```

In situations where the control's content has one directionality but the placeholder needs to have a different directionality, Unicode's bidirectional-algorithm formatting characters can be used in the attribute value:

```
<input name=t1 type=tel placeholder="&#x202B; 1 رقم الهاتف &#x202E;">
<input name=t2 type=tel placeholder="&#x202B; 2 رقم الهاتف &#x202E;">
```

For slightly more clarity, here's the same example using numeric character references instead of inline Arabic:

```
<input name=t1 type=tel
placeholder="&#x202B;&#1585;&#1602;&#1605; &#1575;&#1604;&#1607;&#1
575;&#1578;&#1601; 1&#x202E;">
<input name=t2 type=tel
placeholder="&#x202B;&#1585;&#1602;&#1605; &#1575;&#1604;&#1607;&#1
575;&#1578;&#1601; 2&#x202E;">
```

#### 4.10.5.4 Common input element APIs

`input.value [ = value ]`

Returns the current value of the form control.

Can be set, to change the value.

Throws an "InvalidStateError" DOMException if it is set to any value other than the empty string when the control is a file upload control.

`input.checked [ = value ]`

Returns the current checkedness of the form control.

Can be set, to change the checkedness.

`input.files [ = files ]`



Returns a FileList object listing the selected files of the form control.

Returns null if the control isn't a file control.

Can be set to a FileList object to change the selected files of the form control. For instance, as the result of a drag-and-drop operation.

`input.valueAsDate [ = value ]`

Returns a Date object representing the form control's value, if applicable; otherwise, returns null.

Can be set, to change the value.

Throws an "InvalidStateError" DOMException if the control isn't date- or time-based.

`input.valueAsNumber [ = value ]`

Returns a number representing the form control's value, if applicable; otherwise, returns NaN.

Can be set, to change the value. Setting this to NaN will set the underlying value to the empty string.

Throws an "InvalidStateError" DOMException if the control is neither date- or time-based nor numeric.

`input.stepUp([ n ])`



`input.stepDown([ n ])`



Changes the form control's value by the value given in the step attribute, multiplied by *n*. The default value for *n* is 1.

Throws "InvalidStateError" DOMException if the control is neither date- or time-based nor numeric, or if the step attribute's value is "any".

`input.list`

Returns the [datalist](#) element indicated by the [list](#) attribute.

#### **`input.showPicker()`**

Shows any applicable picker UI for *input*, so that the user can select a value. (If no picker UI is implemented for the given control, then this method does nothing.)

Throws an ["InvalidStateError"](#) [DOMException](#) if *input* is not [mutable](#).

Lance un ["NotAllowedError"](#) [DOMException](#) s'il est appelé sans [activation de l'utilisateur transitoire](#) .

Lève un ["SecurityError"](#) [DOMException](#) si *l'entrée* est à l'intérieur d'une origine croisée [iframe](#), sauf si *l'entrée* est dans les états [Téléchargement de fichier](#) ou [Couleur](#) .

L' [value](#) attribut IDL permet aux scripts de manipuler la [valeur](#) d'un [input](#) élément. L'attribut est dans l'un des modes suivants, qui définissent son comportement :

#### ***valeur***

Lors de l'obtention, retourne la [valeur](#) actuelle de l'élément.

Au réglage :

1. Soit *oldValue* la [valeur](#) de l'élément .
2. Définissez la [valeur](#) de l'élément sur la nouvelle valeur.
3. Définissez l' [indicateur de valeur sale](#) de l'élément sur true.
4. Appelez l' [algorithme de nettoyage de valeur](#) , si l' [type](#) état actuel de l'attribut de l'élément en définit un.
5. [Si la valeur](#) de l'élément (après application de l' [algorithme de nettoyage de valeur](#) ) est différente de *oldValue* et que l'élément a une [position de curseur de saisie de texte](#) , déplacez la [position du curseur de saisie de texte](#) à la fin du contrôle de texte, en désélectionnant tout texte sélectionné et en [réinitialisant la direction de sélection](#) à " none".

#### ***défaut***

Lors de l'obtention, si l'élément a un [value](#) attribut de contenu, renvoie la valeur de cet attribut ; sinon, renvoie la chaîne vide.

Lors du paramétrage, définissez la valeur de l' [value](#) attribut de contenu de l'élément sur la nouvelle valeur.

#### ***par défaut/activé***

Lors de l'obtention, si l'élément a un [value](#) attribut de contenu, renvoie la valeur de cet attribut ; sinon, renvoie la chaîne " on".

Lors du paramétrage, définissez la valeur de l' [value](#) attribut de contenu de l'élément sur la nouvelle valeur.

### **nom de fichier**

A l'obtention, retourne la chaîne " C:\fakepath\" suivi du nom du premier fichier de la liste des [fichiers sélectionnés](#) , s'il y en a un, ou la chaîne vide si la liste est vide.

Au réglage, si la nouvelle valeur est la chaîne vide, vide la liste des [fichiers sélectionnés](#) ; sinon, jetez un " [InvalidStateError](#)" [DOMException](#) .

*Cette exigence de "fakepath" est un triste accident de l'histoire. Voir [l'exemple dans la section État de téléchargement de fichier](#) pour plus d'informations. Étant donné que [les composants de chemin](#) ne sont pas autorisés dans les noms de fichiers de la liste des [fichiers sélectionnés](#) , le "fakepath\" ne peut pas être confondu avec un composant de chemin.*

---

L' [checked](#) attribut IDL permet aux scripts de manipuler la [vérification](#) d'un [input](#) élément. Lors de l'obtention, il doit renvoyer la [vérification](#) actuelle de l'élément ; et lors de la définition, il doit définir la [vérification](#) de l'élément sur la nouvelle valeur et définir l' [indicateur de vérification sale](#) de l'élément sur vrai.

---

L' [files](#) attribut IDL permet aux scripts d'accéder aux [fichiers sélectionnés](#) de l'élément .

Lors de l'obtention, si l'attribut IDL [s'applique](#) , il doit renvoyer un [FileList](#) objet qui représente les [fichiers actuellement sélectionnés](#) . Le même objet doit être retourné jusqu'à ce que la liste des [fichiers sélectionnés](#) change. Si l'attribut IDL [ne s'applique pas](#) , il doit à la place renvoyer null. [\[FILEAPI\]](#)

Lors du réglage, il doit exécuter ces étapes :

1. Si l'attribut IDL [ne s'applique pas](#) ou si la valeur donnée est nulle, alors retournez.
2. Remplacez les [fichiers sélectionnés](#) de l'élément par la valeur donnée.

---

L' **valueAsDate**attribut IDL représente la [valeur](#) de l'élément, interprétée comme une date.

Lors de l'obtention, si l' **valueAsDate**attribut [ne s'applique pas](#) , tel que défini pour l' état actuel de l'attribut **input**de l'élément **type**, renvoie null. Sinon, exécutez l' [algorithme pour convertir une chaîne en un Dateobjet](#) défini pour cet état à la [valeur](#) de l'élément ; si l'algorithme a retourné un **Date**objet, alors retournez-le, sinon, retournez null.

Lors de la définition, si l' **valueAsDate**attribut [ne s'applique pas](#) , tel que défini pour l' état actuel de l'attribut **input**de l'élément **type**, lancez un **"InvalidStateError" DOMException** ; sinon, si la nouvelle valeur n'est pas nulle et n'est pas un **Date**objet, lance une **TypeError**exception ; sinon, si la nouvelle valeur est nulle ou un **Date**objet représentant la valeur de temps NaN, définissez la [valeur](#) de l'élément sur la chaîne vide ; sinon, exécutez l' [algorithme pour convertir un Dateobjet en une chaîne](#) , comme défini pour cet état, sur la nouvelle valeur, et définissez la [valeur](#) de l'élément sur la chaîne résultante.

---

L' **valueAsNumber**attribut IDL représente la [valeur](#) de l'élément, interprétée comme un nombre.

Lors de l'obtention, si l' **valueAsNumber**attribut [ne s'applique pas](#) , tel que défini pour l' état actuel de l'attribut **input**de l'élément **type**, renvoie une valeur Not-a-Number (NaN). Sinon, exécutez l' [algorithme pour convertir une chaîne en un nombre](#) défini pour cet état à la [valeur](#) de l'élément ; si l'algorithme a renvoyé un nombre, renvoyez-le, sinon, renvoyez une valeur Not-a-Number (NaN).

Lors du réglage, si la nouvelle valeur est infinie, lancez une **TypeError**exception. Sinon, si l' **valueAsNumber**attribut [ne s'applique pas](#) , tel que défini pour l' état actuel de l'attribut **input**de l'élément **type**, lancez un **"InvalidStateError" DOMException** . Sinon, si la nouvelle valeur est une valeur Not-a-Number (NaN), définissez la [valeur](#) de l'élément sur la chaîne vide. Sinon, exécutez l' [algorithme pour convertir un nombre en une chaîne](#) , comme défini pour cet état, sur la nouvelle valeur, et définissez la [valeur](#) de l'élément sur la chaîne résultante.

---

Les méthodes et , lorsqu'elles sont invoquées, doivent exécuter l'algorithme suivant : `stepDown (n)` `stepUp (n)`

1. Si les méthodes `stepDown ()` et `ne s'appliquent pas` , comme défini pour l' état actuel de l'attribut de l'élément , lancez alors un `" ".stepUp () inputtypeInvalidStateError DOMException`
2. Si l'élément n'a pas `de valeur autorisée step` , lancez un `" InvalidStateError" DOMException` .
3. Si l'élément a un `minimum` et un `maximum` et que le `minimum` est supérieur au `maximum` , alors retournez.
4. Si l'élément a un `minimum` et un `maximum` et qu'il n'y a pas de valeur supérieure ou égale au `minimum` de l'élément et inférieure ou égale au `maximum` de l'élément qui, une fois soustraite de l' `étape base` , est un multiple entier de la `valeur autorisée step` , puis reviens.
5. Si l'application de l' `algorithme pour convertir une chaîne en un nombre` à la chaîne donnée par la `valeur` de l'élément n'entraîne pas d'erreur, alors laissez *la valeur* être le résultat de cet algorithme. Sinon, laissez *la valeur* être zéro.
6. Soit *valueBeforeStepping* être *value* .
7. Si *la valeur* soustraite de la `base de pas` n'est pas un multiple entier de la `valeur autorisée pas` , alors définissez *la valeur* sur la valeur la plus proche qui, lorsqu'elle est soustraite de la `base de pas` , est un multiple entier de la `valeur autorisée pas` , et qui est inférieure à *la valeur* si la méthode invoquée était la `stepDown ()` méthode, et plus que *la valeur* sinon.

Sinon ( *la valeur* soustraite de la `base de pas` est un multiple entier de la `valeur autorisée pas` ):

1. Soit *n* l'argument.
2. Soit *delta* le pas `de valeur autorisé` multiplié par *n* .
3. Si la méthode invoquée était la `stepDown ()` méthode, inversez *delta* .
4. Soit *value* le résultat de l'ajout de *delta* à *value* .
8. Si l'élément a un `minimum` et que *la valeur* est inférieure à ce `minimum` , définissez *la valeur* sur la plus petite valeur qui, lorsqu'elle est soustraite de la `base de l'étape` , est un multiple entier de la `valeur autorisée step` , et qui est supérieure ou égale au *minimum* .
9. Si l'élément a un `maximum` et que *la valeur* est supérieure à ce `maximum` , définissez *la valeur* sur la plus grande valeur qui, lorsqu'elle est soustraite de la `base de pas` , est un multiple entier de la `valeur autorisée pas` , et qui est inférieure ou égale au *maximum* .

10. Si la méthode invoquée était la `stepDown()` méthode et *la valeur* est supérieure à *valueBeforeStepping*, ou si la méthode invoquée était la `stepUp()` méthode et *la valeur* est inférieure à *valueBeforeStepping*, alors retournez.

Cela garantit que l'appel de la `stepUp()` méthode sur l' `input` élément dans l'exemple suivant ne modifie pas la *valeur* de cet élément :

```
<input type=number value=1 max=0>
```

11. Soit *value as string* le résultat de l'exécution de l' [algorithme pour convertir un nombre en string](#), tel que défini pour l'état actuel de l'attribut `input` de l'élément `type`, sur *value*.
12. Définissez la *valeur* de l'élément sur *value as string*.
- 

L' `list` attribut IDL doit renvoyer l' [élément source des suggestions](#) actuelles, le cas échéant, ou null dans le cas contraire.

---



Les `showPicker()` étapes de la méthode sont :

1. Si `ce` n'est pas [modifiable](#), lancez un `"InvalidStateError" DOMException`.
2. Si l' [origine](#) de [cet objet de paramètres pertinent](#) n'est pas [la même origine](#) que l'[origine de niveau supérieur](#) de `cet` objet [de paramètres pertinent](#) et que `cet` attribut n'est pas dans l'état [de téléchargement de fichier](#) ou dans l'état [de couleur](#), lancez un `"".typeSecurityError DOMException`

*Les entrées Fichier et Couleur sont exemptées de cette vérification pour des raisons historiques : leur [comportement d'activation des entrées](#) montre également leurs sélecteurs et n'a jamais été protégé par une vérification d'origine.*

3. Si `cet` objet [global pertinent](#) n'a pas d' [activation transitoire](#), lancez alors un `"NotAllowedError" DOMException`.



4. [Montrez le sélecteur, le cas échéant](#) , pour [ce](#) .

Pour **afficher le sélecteur, le cas échéant** pour un élément [input](#) element :

1. Si [l'objet global pertinent](#) de l'élément n'a pas [d'activation transitoire](#) , alors retournez.
2. Si *element* n'est pas [mutable](#) , alors retournez.
3. Si l'attribut de l'élément [type](#) est à l'état [Téléchargement de fichier](#) , exécutez ces étapes [en parallèle](#) :
  1. En option, attendez que toute exécution précédente de cet algorithme soit terminée.
  2. Affichez une invite à l'utilisateur lui demandant de spécifier certains fichiers. Si l'[multiple](#)attribut n'est pas défini sur l'élément , il ne doit pas y avoir plus d'un fichier sélectionné ; sinon, n'importe quel nombre peut être sélectionné. Les fichiers peuvent provenir du système de fichiers ou être créés à la volée, par exemple une photo prise à partir d'un appareil photo connecté à l'appareil de l'utilisateur.
  3. Attendez que l'utilisateur ait fait sa sélection.
  4. Si l'utilisateur a rejeté l'invite sans modifier sa sélection, mettez en file d'attente [une tâche d'élément](#) sur la [source de tâche d'interaction utilisateur](#) donnée *element* pour [déclencher un événement](#) nommé [cancel](#) à *element* , avec l'[bubbles](#)attribut initialisé à true.
  5. Sinon, [mettez à jour la sélection de fichiers](#) pour l'élément .

*Comme pour toutes les spécifications d'interface utilisateur, les agents utilisateurs ont une grande liberté dans la façon dont ils interprètent ces exigences. Le texte ci-dessus implique qu'un utilisateur rejette l'invite ou modifie sa sélection ; exactement l'un d'entre eux sera vrai. Mais le mappage de ces possibilités à des éléments d'interface utilisateur spécifiques n'est pas mandaté par la norme. Par exemple, un agent utilisateur peut interpréter le fait de cliquer sur le bouton "Annuler" lorsque des fichiers ont été précédemment sélectionnés comme un changement de sélection pour sélectionner zéro fichier, déclenchant ainsi [input](#) et [change](#). Ou il peut interpréter un tel clic comme un rejet qui laisse la sélection inchangée, déclenchant ainsi [cancel](#). De même, c'est à l'agent utilisateur de décider si la re-sélection des mêmes fichiers compte que ceux précédemment sélectionnés compte comme un rejet ou comme un changement de sélection.*

4. Sinon, l'agent utilisateur doit afficher toute interface utilisateur pertinente pour sélectionner une valeur pour *element* , comme il le ferait normalement lorsque l'utilisateur interagit avec le contrôle. (Si aucune interface utilisateur de ce type ne s'applique à *element* , cette étape ne fait rien.)

Si une telle interface utilisateur est affichée, elle doit respecter les exigences énoncées dans les parties pertinentes de la spécification concernant le comportement *de l'élément* compte tenu de son type état d'attribut. (Par exemple, diverses sections décrivent les restrictions sur la chaîne de valeur résultante .)

Cette étape peut avoir des effets secondaires, tels que la fermeture d'autres sélecteurs précédemment affichés par cet algorithme. (Si cela ferme un sélecteur de sélection de fichiers, alors selon ce qui précède, cela conduira à déclencher soit input et change des événements, soit un cancel événement.)

*Au moment d'écrire ces lignes, les implémentations de navigateur typiques affichent une telle interface utilisateur de sélecteur pour :*

1. input les éléments dont type les attributs sont dans les états Date , Month , Week , Time , Local Date and Time et Color ;
2. input les éléments dans divers états qui ont un élément source de suggestions ; et
3. input les éléments dont type l'attribut est dans l' état Téléchargement de fichier (bien que ceux-ci soient gérés via le cas spécial ci-dessus, au lieu de cette étape).

*Cependant, le but de cette étape est de déclencher toute implémentation de l'interface utilisateur du sélecteur. Ainsi, par exemple, si un agent utilisateur implémentait une interface utilisateur de sélection de mot de passe pour l' état Mot de passe , cette méthode devrait alors afficher cette interface utilisateur de sélection lorsqu'elle est appelée sur une entrée de mot de passe.*

#### 4.10.5.5 Comportements d'événements courants

Lorsque les événements input et s'appliquent (ce qui est le cas pour tous les contrôles autres que les boutons et ceux dont l' attribut est à l' état Masqué ), les événements sont déclenchés pour indiquer que l'utilisateur a interagi avec le contrôle. L' événement se déclenche chaque fois que l'utilisateur a modifié les données du contrôle. L' événement se déclenche lorsque la valeur est validée, si cela a du sens pour le contrôle, ou bien lorsque le contrôle perd le focus . Dans tous les cas, l' événement précède l' événement correspondant (le cas échéant). changeinputtypeinputchangeinputchange

Lorsqu'un input élément a un comportement d'activation d'entrée défini , les règles de distribution de ces événements, si elles s'appliquent , sont données dans la section ci-dessus qui définit l' type état de l'attribut. (C'est le cas pour tous input les contrôles dont l' type attribut est à l' état Case à cocher , à l' état Bouton radio ou à l' état Téléchargement de fichier .)

Pour input les éléments sans comportement d'activation d'entrée défini, mais auxquels ces événements s'appliquent, et pour lesquels l'interface utilisateur implique à la fois une manipulation interactive et une action de validation explicite, lorsque l'utilisateur modifie la valeur de l'élément, l'agent utilisateur doit mettre en file d'attente une tâche d'élément sur la source de tâche d'interaction utilisateur donnée à l'input élément pour déclencher un événement nommé input sur l'input élément, avec les attributs bubbles et composed initialisés à vrai, et chaque fois que l'utilisateur valide la modification, l'agent utilisateur doit mettre en file d'attente une tâche d'élément sur la source de tâche d'interaction utilisateur donné l'input élément pour déclencher un événement nommé change à l'input élément, avec l'bubbles attribut initialisé à true.

Un exemple d'interface utilisateur impliquant à la fois une manipulation interactive et une action de validation serait un contrôle Range qui utilise un curseur, lorsqu'il est manipulé à l'aide d'un dispositif de pointage. Pendant que l'utilisateur fait glisser le bouton du contrôle, input les événements se déclenchent chaque fois que la position change, tandis que l'change événement ne se déclenche que lorsque l'utilisateur lâche le bouton, s'engageant sur une valeur spécifique.

Pour input les éléments sans comportement d'activation d'entrée défini, mais auxquels ces événements s'appliquent, et pour lesquels l'interface utilisateur implique une action de validation explicite mais aucune manipulation intermédiaire, alors chaque fois que l'utilisateur valide une modification de la valeur de l'élément, l'agent utilisateur doit mettre en file d'attente une tâche d'élément sur la source de la tâche d'interaction utilisateur étant donné l'input élément pour déclencher d'abord un événement nommé input à l'input élément, avec les attributs bubbles et composed initialisés à true, puis déclencher un événement nommé change à l'input élément, avec le bubbles attribut initialisé à true.

Un exemple d'interface utilisateur avec une action de validation serait un contrôle de couleur composé d'un seul bouton qui fait apparaître une roue chromatique : si la valeur ne change que lorsque la boîte de dialogue est fermée, il s'agirait alors de l'action de validation explicite. D'autre part, si la manipulation du contrôle modifie la couleur de manière interactive, il se peut qu'il n'y ait aucune action de validation. Un autre exemple d'interface utilisateur avec une action de validation serait un contrôle de date qui permet à la fois la saisie de texte par l'utilisateur et la sélection de l'utilisateur à partir d'un calendrier déroulant : bien que la saisie de texte puisse ne pas avoir d'étape de validation explicite, la sélection d'une date dans la liste déroulante calendrier vers le bas, puis rejeter la liste déroulante serait une action de validation.

Pour input les éléments sans comportement d'activation d'entrée défini, mais auxquels ces événements s'appliquent, chaque fois que l'utilisateur modifie la valeur de l'élément sans action de validation explicite, l'agent utilisateur doit mettre en file d'attente une tâche d'élément sur la source de la tâche d'interaction de l'utilisateur compte tenu de l'input élément à déclenche un événement nommé input au niveau de l'input élément, avec les attributs bubbles et composed initialisés à true. L'événement

correspondant [change](#), le cas échéant, sera déclenché lorsque le contrôle [perd le focus](#) .

Les exemples d'un utilisateur modifiant la [valeur](#) de l'élément incluent l'utilisateur tapant dans un contrôle de texte, collant une nouvelle valeur dans le contrôle ou annulant une modification dans ce contrôle. Certaines interactions de l'utilisateur ne modifient pas la valeur, par exemple, appuyer sur la touche "supprimer" dans un contrôle de texte vide ou remplacer du texte dans le contrôle par du texte du presse-papiers qui se trouve être exactement le même texte.

Un contrôle [Range](#) sous la forme d'un curseur sur lequel l'utilisateur s'est [concentré](#) et avec lequel il interagit à l'aide d'un clavier serait un autre exemple de l'utilisateur modifiant la [valeur](#) de l'élément sans étape de validation.

Dans le cas de [tâches](#) qui ne font que déclencher un [input](#) événement, les agents utilisateurs peuvent attendre une pause appropriée dans l'interaction de l'utilisateur avant de [mettre](#) les tâches en file d'attente ; par exemple, un agent utilisateur pourrait attendre que l'utilisateur n'ait pas appuyé sur une touche pendant 100 ms, afin de ne déclencher l'événement que lorsque l'utilisateur fait une pause, au lieu de le faire en continu à chaque frappe.

Lorsque l'agent utilisateur doit modifier la [valeur](#) [input](#) d'un élément au nom de l'utilisateur (par exemple dans le cadre d'une fonctionnalité de préremplissage de formulaire), l'agent utilisateur doit [mettre en file d'attente une tâche d'élément](#) sur la [source de tâche d'interaction utilisateur](#) étant donné que l'élément doit d'abord mettre à jour la [valeur](#) en conséquence, puis [déclenchez un événement](#) nommé au niveau de l'élément, avec les attributs et initialisés à true, puis [déclenchez un événement](#) nommé au niveau de l'élément, avec l'attribut initialisé à true. [inputinputinputbubblescomposedchangeinputbubbles](#)

*Ces événements ne sont pas déclenchés en réponse aux modifications apportées aux valeurs des contrôles de formulaire par les scripts. (Cela facilite la mise à jour des valeurs des contrôles de formulaire en réponse à la manipulation des contrôles par l'utilisateur, sans avoir à filtrer ensuite les propres modifications du script pour éviter une boucle infinie.)*

*Ces événements ne sont pas non plus déclenchés lorsque le navigateur modifie les valeurs des contrôles de formulaire dans le cadre de [la restauration de l'état pendant la navigation](#) .*

#### 4.10.6 L' [button](#) élément



##### [Catégories](#) :

[Contenu du flux](#) .

[Contenu de la phrase](#) .

[Contenu interactif](#) .

[Listed](#) , [labelable](#) , [submittable](#) , and [autocapitalize-heriting form-associated element](#) .

[Contenu palpable](#) .

### **Contextes dans lesquels cet élément peut être utilisé :**

Où [le contenu du phrasé](#) est attendu.

### **Modèle de contenu :**

[Phrasing content](#) , mais il ne doit y avoir aucun descendant [de contenu interactif](#) et aucun descendant avec l' [tabindex](#) attribut spécifié.

### **Omission de balise dans text/html :**

Aucune des deux balises n'est omise.

### **Attributs de contenu :**

#### Attributs globaux

[disabled](#)— Si le contrôle du formulaire est désactivé

[form](#)— Associe l'élément à un [form](#)élément

[formaction](#)— [URL](#) à utiliser pour [la soumission du formulaire](#)

[formenctype](#)— Type d'encodage [de la liste d'entrées](#) à utiliser pour [la soumission du formulaire](#)

[formmethod](#)— Variante à utiliser pour [la soumission de formulaire](#)

[formnovalidate](#)— Contourner la validation du contrôle du formulaire pour [la soumission du formulaire](#)

[formtarget](#)— [Navigable](#) pour [la soumission du formulaire](#)

[name](#)— Nom de l'élément à utiliser pour [la soumission du formulaire](#) et dans l' [form.elements](#)API

[popoverhidetarget](#)

[popovershowtarget](#)

[popovertoggletarget](#)

[type](#)— Type de bouton

[value](#)— Valeur à utiliser pour [la soumission du formulaire](#)

### **Considérations d'accessibilité :**

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### **Interface DOM :**

```
[Exposed=Window]
```

```
interface HTMLButtonElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute boolean disabled;
```

readonly attribute <a href="#">HTMLFormElement?</a> <a href="#">form</a> ;
[ <a href="#">CEReactions</a> ] attribute USVString <a href="#">formAction</a> ;
[ <a href="#">CEReactions</a> ] attribute DOMString <a href="#">formEnctype</a> ;
[ <a href="#">CEReactions</a> ] attribute DOMString <a href="#">formMethod</a> ;
[ <a href="#">CEReactions</a> ] attribute boolean <a href="#">formNoValidate</a> ;
[ <a href="#">CEReactions</a> ] attribute DOMString <a href="#">formTarget</a> ;
[ <a href="#">CEReactions</a> ] attribute DOMString <a href="#">name</a> ;
[ <a href="#">CEReactions</a> ] attribute DOMString <a href="#">type</a> ;
[ <a href="#">CEReactions</a> ] attribute DOMString <a href="#">value</a> ;
readonly attribute boolean <a href="#">willValidate</a> ;
readonly attribute <a href="#">ValidityState</a> <a href="#">validity</a> ;
readonly attribute DOMString <a href="#">validationMessage</a> ;
boolean <a href="#">checkValidity</a> ();
boolean <a href="#">reportValidity</a> ();
undefined <a href="#">setCustomValidity</a> (DOMString error);
readonly attribute <a href="#">NodeList</a> <a href="#">labels</a> ;
};
<a href="#">HTMLButtonElement</a> includes <a href="#">PopoverTargetElement</a> ;

L' **button** élément [représente](#) un bouton étiqueté par son contenu.

L'élément est un [bouton](#) .

L' **type** attribut contrôle le comportement du bouton lorsqu'il est activé. C'est un [attribut énuméré](#) . Le tableau suivant répertorie les mots-clés et les états de

l'attribut — les mots-clés de la colonne de gauche correspondent aux états de la cellule de la deuxième colonne sur la même ligne que le mot-clé.

Mot-clé	État	Brève description
<code>submit</code>	<a href="#">Bouton de soumission</a>	Soumet le formulaire.
<code>reset</code>	<a href="#">Bouton de réinitialisation</a>	Réinitialise le formulaire.
<code>button</code>	<a href="#">Bouton</a>	Ne fait rien.

La [valeur par défaut manquante](#) et [la valeur par défaut non valide](#) correspondent à l'état [du bouton Soumettre](#).

Si l'attribut `type` est à l'état [Bouton d'envoi](#), l'élément est spécifiquement un [bouton d'envoi](#).

**Validation de contrainte** : si l'attribut `type` est dans l'état [Bouton de réinitialisation](#) ou dans l'état [Bouton](#), l'élément est [exclu de la validation de contrainte](#).

[Le comportement d'activation](#) d'un élément `button` est :

1. Si l'élément est [disabled](#), alors revenez.
2. Si le [document de nœud](#) de l'élément n'est pas [complètement actif](#), alors retournez.
3. Si l'élément a un [propriétaire de formulaire](#), activez l'état de l'attribut de l'élément `type`, puis :

#### ***Bouton de soumission***

[Soumettre le propriétaire](#) du formulaire de l'élément à partir de l'élément.

#### ***Bouton de réinitialisation***

[Réinitialiser le propriétaire](#) du formulaire de l'élément.

#### ***Bouton***

Ne fait rien.

4. Exécutez le [comportement d'activation de l'attribut cible popover](#) élément donné.

L'attribut `form` est utilisé pour associer explicitement l'élément `button` à son [propriétaire de formulaire](#). L'attribut `name` représente le nom de l'élément. L'attribut `disabled` est utilisé pour rendre le contrôle non interactif et pour empêcher la soumission de sa valeur. Les attributs `formaction`, `formenctype`, et sont `formmethod` des [attributs de soumission de formulaire](#). `formnovalidate` `formtarget`



*L' [formnovalidate](#) attribut peut être utilisé pour créer des boutons de soumission qui ne déclenchent pas la validation de la contrainte.*

Les [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), et [formtarget](#) ne doivent pas être spécifiés si l' [type](#) attribut de l'élément n'est pas dans l' état [Soumettre le bouton](#) .

L' **value** attribut donne la valeur de l'élément pour les besoins de la soumission du formulaire. La [valeur](#) de l'élément est la valeur de l' [value](#) attribut de l'élément, s'il y en a un, ou la chaîne vide sinon.

*Un bouton (et sa valeur) n'est inclus dans la soumission du formulaire que si le bouton lui-même a été utilisé pour lancer la soumission du formulaire.*

---

L' **value** attribut IDL doit [refléter](#) l'attribut de contenu du même nom.

L' **type** attribut IDL doit [refléter](#) l'attribut de contenu du même nom, [limité aux seules valeurs connues](#) .

Les attributs [willValidate](#), [validity](#) et [validationMessage](#) IDL et [checkValidity\(\)](#) les méthodes , et font partie de l' [API de validation de contrainte](#) . L' attribut IDL fournit une liste des s de l'élément. Les attributs , et IDL font partie de l'API des formulaires de l'élément. [reportValidity\(\)](#) [setCustomValidity\(\)](#) [labels](#) [labeldisabled](#) [formname](#)

Le bouton suivant s'intitule "Afficher l'indice" et ouvre une boîte de dialogue lorsqu'il est activé :

```
<button type=button
  onclick="alert('This 15-20 minute piece was composed by
George Gershwin.')">
  Show hint
</button>
```

#### 4.10.7 L' **select** élément





## Catégories :

[Contenu du flux](#) .

[Contenu de la phrase](#) .

[Contenu interactif](#) .

[Listed](#) , [labelable](#) , [submittable](#) , [resettable](#) et [autocapitalize-heriting form-associated element](#) .

[Contenu palpable](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

## Modèle de contenu :

Zéro ou plusieurs éléments [option](#), [optgroup](#) et [de support de script](#) .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

[Attributs globaux](#)

[autocomplete](#)- Astuce pour la fonction de remplissage automatique du formulaire

[disabled](#)— Si le contrôle du formulaire est désactivé

[form](#)— Associe l'élément à un [form](#)élément

[multiple](#)— S'il faut autoriser plusieurs valeurs

[name](#)— Nom de l'élément à utiliser pour [la soumission du formulaire](#) et dans l' [form.elements](#)API

[required](#)— Si le contrôle est requis pour [la soumission du formulaire](#)

[size](#)— Taille du contrôle

## Considérations d'accessibilité :

Si l'élément a un [multiple](#)attribut ou un [size](#)attribut avec une valeur > 1 : [pour les auteurs](#) ; [pour les exécutants](#) .

Sinon : [pour les auteurs](#) ; [pour les exécutants](#) .

## Interface DOM :

[Exposed=Window]

```
interface HTMLSelectElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute DOMString autocomplete;
```

```
    [CEReactions] attribute boolean disabled;
```

```
    readonly attribute HTMLFormElement? form;
```

[[CEReactions](#)] attribute boolean [multiple](#);

[[CEReactions](#)] attribute DOMString [name](#);

[[CEReactions](#)] attribute boolean [required](#);

[[CEReactions](#)] attribute unsigned long [size](#);

readonly attribute DOMString [type](#);

[SameObject] readonly attribute [HTMLOptionsCollection](#)  
[options](#);

[[CEReactions](#)] attribute unsigned long [length](#);

getter [HTMLOptionElement](#)? [item](#)(unsigned long index);

[HTMLOptionElement](#)? [namedItem](#)(DOMString name);

[[CEReactions](#)] undefined [add](#)(([HTMLOptionElement](#) or  
[HTMLOptGroupElement](#)) element, optional ([HTMLElement](#) or  
long)? before = null);

[[CEReactions](#)] undefined [remove](#)(); // ChildNode overload

[[CEReactions](#)] undefined [remove](#)(long index);

[[CEReactions](#)] [setter](#) undefined (unsigned long index,  
[HTMLOptionElement](#)? option);

[SameObject] readonly attribute [HTMLCollection](#)  
[selectedOptions](#);

attribute long [selectedIndex](#);

attribute DOMString [value](#);

```

readonly attribute boolean willValidate;

readonly attribute ValidityState validity;

readonly attribute DOMString validationMessage;

boolean checkValidity();

boolean reportValidity();

undefined setCustomValidity(DOMString error);

readonly attribute NodeList labels;

};

```

L' [select](#) élément représente une commande de sélection parmi un ensemble d'options.



L' **multiple** attribut est un [attribut booléen](#) . Si l'attribut est présent, alors l' [select](#) élément [représente](#) un contrôle pour sélectionner zéro ou plusieurs options dans la [liste des options](#) . Si l'attribut est absent, alors l' [select](#) élément [représente](#) un contrôle pour sélectionner une seule option dans la [liste des options](#) .



L' **size** attribut donne le nombre d'options à montrer à l'utilisateur. L' [size](#) attribut, s'il est spécifié, doit avoir une valeur qui est un [entier non négatif valide](#) supérieur à zéro.

La **taille d'affichage** d'un [select](#) élément est le résultat de l'application des [règles d'analyse des entiers non négatifs](#) à la valeur de [size](#) l'attribut de l'élément, s'il en a un et que l'analyse est réussie. Si l'application de ces règles à la valeur de l'attribut ne réussit pas, ou si l' [size](#) attribut est absent, alors la [taille d'affichage](#) de l'élément est de 4 si l' [multiple](#) attribut de contenu de l'élément est présent, et de 1 sinon.

La **liste des options** d'un [select](#) élément se compose de tous les [option](#) éléments enfants de l' [select](#) élément et de tous les [option](#) éléments enfants de tous les [optgroup](#) éléments enfants de l' [select](#) élément, dans [l'ordre de l'arborescence](#) .

L' **required** attribut est un [attribut booléen](#) . Lorsque spécifié, l'utilisateur devra sélectionner une valeur avant de soumettre le formulaire.

Si un [select](#) élément a un [required](#) attribut spécifié, n'a pas d' [multiple](#) attribut spécifié et a une [taille d'affichage](#) de 1 ; et si la [valeur](#) du premier [option](#) élément dans la [liste d'options de select](#) l'élément (le cas échéant) est la chaîne vide et que le nœud parent de cet élément est l' élément (et non un élément), alors il s'agit de l' **option d'étiquette d'espace réservé** de l'élément `.optionselectoptgroupoptionselect`

Si un [select](#) élément a un [required](#) attribut spécifié, n'a pas d' [multiple](#) attribut spécifié et a une [taille d'affichage](#) de 1, alors l' [select](#) élément doit avoir une [option d'étiquette d'espace réservé](#) .

*En pratique, l'exigence énoncée dans le paragraphe ci-dessus ne peut s'appliquer que lorsqu'un [select](#) élément n'a pas d' [size](#) attribut de valeur supérieure à 1.*

**Validation de contrainte** : si l'attribut de l'élément est [required](#) spécifié, et si aucun des [option](#) éléments de la [liste d'options select](#) de l'élément n'a sa [sélection](#) définie sur true, ou si le seul élément de la [liste d'options de](#) l'élément dont [la sélection](#) est définie sur true est l' [étiquette d'espace réservé option](#) , alors l'élément [souffre d'être manquant](#) `.optionselect`

Si l' [multiple](#) attribut est absent et que l'élément n'est pas [désactivé](#) , alors l'agent utilisateur devrait permettre à l'utilisateur de choisir un [option](#) élément dans sa [liste d'options](#) qui n'est pas lui-même [désactivé](#) . **Lors de la sélection** de cet [option](#) élément (soit par un clic, soit en désactivant l'élément après avoir modifié sa valeur, soit via une [commande de menu](#) , soit via tout autre mécanisme), et avant que l'événement d'interaction utilisateur concerné ne soit mis en file d'attente (par exemple avant l' événement), l'agent utilisateur doit définir la [sélection](#) de l'élément sélectionné sur vrai, définir sa [saleté](#) sur vrai, puis [envoyerclickoptionselectavis de mise à jour](#) .

Si l' [multiple](#) attribut est absent, chaque fois qu'un [option](#) élément de la [liste d'options select](#) de l'élément a sa [sélection](#) définie sur true, et chaque fois qu'un élément dont [la sélection](#) est définie sur true est ajouté à la [liste d'options](#) de l'élément , l'agent utilisateur doit définir la [sélection](#) de tous les autres éléments de sa [liste d'options](#) à false. `optionselectoption`

Si l' [multiple](#) attribut est absent et que [la taille d'affichage](#) de l'élément est supérieure à 1, alors l'agent utilisateur devrait également permettre à l'utilisateur de demander que le [option](#) dont [la sélection](#) est vraie, le cas échéant, soit désélectionné. Lorsque cette demande est transmise à l'agent utilisateur, et avant que l'événement d'interaction utilisateur pertinent ne soit mis en file d'attente (par exemple avant l' [click](#) événement), l'agent utilisateur doit définir la [sélection](#) de

cet optionélément sur faux, définir sa saleté sur vrai, puis envoyer select des notifications de mise à jour .

L' **algorithme de réglage de la sélection** , étant donné un élémentselect element , consiste à exécuter les étapes suivantes :

1. Si l'attribut de l' élémentmultiple est absent et que la taille d'affichage de l'élément est de 1, et qu'aucun élément de la liste d'options de l' élément n'a sa sélection définie sur vrai, alors définissez la sélection du premier élément de la liste des options dans l'arborescence order qui n'est pas disabled , le cas échéant, à true et return.optionoption
2. Si l'attribut de l' élémentmultiple est absent et que deux optionéléments ou plus dans la liste d'options de l' élément ont leur sélection définie sur vrai, alors définissez la sélection de tous les éléments sauf le dernier avec sa sélection définie sur vrai dans la liste des options de l' arborescence commande à faux.option

Les option étapes d'insertion de l'élément HTML , étant donné *insertNode* , sont :

1. Si le parent de *insertNode*select est un élément, ou si le parent de *insertNode*optgroup est un élément dont le parent est un selectélément, alors exécutez l'algorithme de réglage de sélectionselect de cet élément .

Les option étapes de suppression de l'élément HTML , étant donnés *removeNode* et *oldParent* , sont :

1. Si *oldParent* est un selectélément, ou *oldParent* est un optgroupélément dont le parent est un selectélément, alors exécutez l' algorithme de réglage de sélectionselect de cet élément .

Si un optionélément de la liste des options **demande une réinitialisation** , exécutez l'algorithme de réglage de sélectionselect de cet élément .

Si l' multipleattribut est présent et que l'élément n'est pas désactivé , alors l'agent utilisateur doit autoriser l'utilisateur à **basculer** la sélection des optionéléments dans sa liste d'options qui ne sont pas elles-mêmes désactivées . Lorsqu'un tel élément est basculé (soit par un clic, soit par une commande de menu , ou tout autre mécanisme), et avant que l'événement d'interaction utilisateur pertinent ne soit mis en file d'attente (par exemple, avant un clickévénement connexe), la sélection de l' optionélément doit être modifiée ( du vrai au faux ou du faux au vrai), la saletéde l'élément doit être défini sur true et l'agent utilisateur doit envoyer select des notifications de mise à jour .

Lorsque l'agent utilisateur doit **envoyer select des notifications de mise à jour** , mettez en file d'attente une tâche d'élément sur la source de tâche d'interaction utilisateur en fonction de l' select élément pour exécuter ces étapes :

1. Lancez un événement nommé `input` au niveau de l' `select` élément, avec les attributs `bubbles` et `composed` initialisés à true.
2. Lancez un événement nommé `change` au niveau de l' `select` élément, avec l' `bubbles` attribut initialisé à true.

L' `algorithm` de réinitialisation pour `select` les éléments consiste à parcourir tous les `option` éléments de la `liste d'options` de l'élément, à définir leur `sélection` sur true si l' `option` élément a un `selected` attribut, et false sinon, à définir leur `saleté` sur false, puis à demander aux `option` éléments `une réinitialisation`.

L' `form` attribut est utilisé pour associer explicitement l' `select` élément à son `propriétaire de formulaire`. L' `name` attribut représente le nom de l'élément. L' `disabled` attribut est utilisé pour rendre le contrôle non interactif et pour empêcher la soumission de sa valeur. L' `autocomplete` attribut contrôle la façon dont l'agent utilisateur fournit le comportement de remplissage automatique.

Un `select` élément qui n'est pas `désactivé` est `mutable`.

`select.type`

✓

Renvoie " `select-multiple`" si l'élément a un `multiple` attribut, et " `select-one`" sinon.

`select.options`

✓

Renvoie une `HTMLOptionsCollection` de la `liste des options`.

`select.length` [ = value ]

Renvoie le nombre d'éléments dans la `liste d'options`.

Lorsqu'il est défini sur un nombre inférieur, tronque le nombre d' `option` éléments dans le fichier `select`.

Lorsqu'il est défini sur un nombre supérieur, ajoute de nouveaux `option` éléments vides au fichier `select`.

`element = select.item(index)`

✓

`select[index]`

Renvoie l'élément avec l' `index` index de la `liste des options`. Les éléments sont triés par `ordre arborescent`.

`element = select.namedItem(name)`

✓

Renvoie le premier élément avec `ID` ou `name` nom de la `liste d'options`.

Renvoie null si aucun élément avec cet `ID` n'a pu être trouvé.

```
select.add(element [, before ])
```

✓

Insère *un élément* avant le nœud donné par *before* .

L' argument *avant* peut être un nombre, auquel cas *element* est inséré avant l'élément portant ce numéro, ou un élément de la [liste d'options](#) , auquel cas *element* est inséré avant cet élément.

Si *avant* est omis, nul ou un nombre hors plage, alors *l'élément* sera ajouté à la fin de la liste.

Cette méthode lancera

un "[HierarchyRequestError](#)" [DOMException](#) si *l'élément* est un ancêtre de l'élément dans lequel il doit être inséré.

```
select.selectedOptions
```

✓

Renvoie une [HTMLCollection](#) de la [liste des options](#) sélectionnées.

```
select.selectedIndex [ = value ]
```

✓

Renvoie l'index du premier élément sélectionné, le cas échéant, ou -1 s'il n'y a pas d'élément sélectionné.

Peut être réglé, pour changer la sélection.

```
select.value [ = value ]
```

Renvoie la [valeur](#) du premier élément sélectionné, le cas échéant, ou la chaîne vide s'il n'y a pas d'élément sélectionné.

Peut être réglé, pour changer la sélection.

L' **type**attribut IDL, à l'obtention, doit retourner la chaîne " `select-one`" si l' [multiple](#)attribut est absent, et la chaîne " `select-multiple`" si l' [multiple](#) attribut est présent.

L' **options**attribut IDL doit renvoyer un [HTMLOptionsCollection](#)enraciné au [select](#)nœud, dont le filtre correspond aux éléments de la [liste des options](#) .

La [options](#)collection se reflète également sur l' [HTMLSelectElement](#)objet. Les [indices de propriété pris en charge](#) à tout instant sont les indices pris en charge par l'objet renvoyé par l' [options](#)attribut à cet instant.

L' **length** attribut IDL doit renvoyer le nombre de nœuds [représentés](#) par la [options](#)collection. Au réglage, il doit agir comme l'attribut du même nom sur la [options](#)collection.

La méthode doit renvoyer la valeur renvoyée par [la méthode du même nom](#) sur la collection, lorsqu'elle est invoquée avec le même argument. [item\(index\)](#) [options](#)

La méthode doit renvoyer la valeur renvoyée par [la méthode du même nom](#) sur la collection, lorsqu'elle est invoquée avec le même argument. `namedItem(name)options`

Lorsque l'agent utilisateur doit [définir la valeur d'une nouvelle propriété indexée](#) ou [définir la valeur d'une propriété indexée existante](#) pour un `select` élément, il doit à la place exécuter [l'algorithme correspondant](#) sur la collection `select` de l'élément `options`.

De même, la méthode doit agir comme sa méthode homonyme sur cette même collection. `add(element, before)options`



La `remove()` méthode doit agir comme sa méthode homonyme sur cette même `options` collection lorsqu'elle a des arguments, et comme sa méthode homonyme sur l' `ChildNode` interface implémentée par l' `HTMLSelectElement` interface ancêtre `Element` lorsqu'elle n'a pas d'arguments.

L' `selectedOptions` attribut IDL doit renvoyer un `HTMLCollection` enraciné au `select` nœud, dont le filtre correspond aux éléments de la [liste des options](#) dont [la sélection](#) est définie sur `true`.

L' `selectedIndex` attribut IDL, lors de l'obtention, doit renvoyer l' [index](#) du premier `option` élément de la [liste des options](#) dans [l'ordre de l'arborescence](#) dont [la sélection](#) est définie sur `true`, le cas échéant. S'il n'y en a pas, alors il doit retourner `-1`.

Lors de la définition, l' `selectedIndex` attribut doit définir la [sélection](#) de tous les `option` éléments de la [liste d'options](#) sur `false`, puis l' `option` élément de la [liste d'options](#) dont l' [index](#) est la nouvelle valeur donnée, le cas échéant, doit avoir sa [sélection](#) définie sur `true` et son [saleté](#) définie sur `true`.

*Il peut en résulter qu'aucun élément n'a une [valeur de sélection](#) définie sur `true` même dans le cas où l' `select` élément n'a pas [multiple](#) d'attribut et une [taille d'affichage](#) de 1.*

L' `value` attribut IDL, lors de l'obtention, doit renvoyer la [valeur](#) du premier `option` élément de la [liste des options](#) dans [l'ordre de l'arborescence](#) dont [la sélection](#) est définie sur `true`, le cas échéant. S'il n'y en a pas, il doit renvoyer la chaîne vide.

Lors de la définition, l' `value` attribut doit définir la [sélection](#) de tous les `option` éléments de la [liste d'options](#) sur `false`, puis le premier `option` élément de la [liste d'options](#), dans [l'ordre de l'arborescence](#), dont [la valeur](#) est égale à la nouvelle valeur donnée, le cas échéant, doit avoir sa [sélection](#) définie sur `true` et sa [saleté](#) définie sur `true`.



*Il peut en résulter qu'aucun élément n'a une valeur de sélection définie sur true même dans le cas où l' select élément n'a pas multiple d'attribut et une taille d'affichage de 1.*

Les attributs **multiple**, **required** et **size** IDL doivent refléter les attributs de contenu respectifs du même nom. L' size attribut IDL a une valeur par défaut de zéro.

*Pour des raisons historiques, la valeur par défaut de l' size attribut IDL ne renvoie pas la taille réelle utilisée, qui, en l'absence de l' size attribut content, est soit 1 soit 4 selon la présence de l' multiple attribut.*

Les attributs willValidate, validity et validationMessage IDL et checkValidity() les méthodes, et font partie de l' API de validation de contrainte. L' attribut IDL fournit une liste des s de l'élément. Les attributs, et IDL font partie de l'API des formulaires de l'élément. reportValidity() setCustomValidity() labels labeldisabled formname

L'exemple suivant montre comment un select élément peut être utilisé pour offrir à l'utilisateur un ensemble d'options parmi lesquelles l'utilisateur peut sélectionner une seule option. L'option par défaut est présélectionnée.

```
<p>
  <label for="unittype">Select unit type:</label>
  <select id="unittype" name="unittype">
    <option value="1"> Miner </option>
    <option value="2"> Puffer </option>
    <option value="3" selected> Snipey </option>
    <option value="4"> Max </option>
    <option value="5"> Firebot </option>
  </select>
</p>
```

Lorsqu'il n'y a pas d'option par défaut, un espace réservé peut être utilisé à la place :

```
<select name="unittype" required>
  <option value=""> Select unit type </option>
  <option value="1"> Miner </option>
  <option value="2"> Puffer </option>
  <option value="3"> Snipey </option>
  <option value="4"> Max </option>
  <option value="5"> Firebot </option>
</select>
```

Ici, l'utilisateur se voit proposer un ensemble d'options parmi lesquelles il peut sélectionner n'importe quel nombre. Par défaut, les cinq options sont sélectionnées.

```
<p>
  <label for="allowedunits">Select unit types to enable on this
  map:</label>
  <select id="allowedunits" name="allowedunits" multiple>
    <option value="1" selected> Miner </option>
    <option value="2" selected> Puffer </option>
    <option value="3" selected> Snipey </option>
    <option value="4" selected> Max </option>
    <option value="5" selected> Firebot </option>
  </select>
</p>
```

Parfois, un utilisateur doit sélectionner un ou plusieurs éléments. Cet exemple montre une telle interface.

```
<label>
  Select the songs from that you would like on your Act II Mix Tape:
  <select multiple required name="act2">
    <option value="s1">It Sucks to Be Me (Reprise)
    <option value="s2">There is Life Outside Your Apartment
    <option value="s3">The More You Ruv Someone
    <option value="s4">Schadenfreude
    <option value="s5">I Wish I Could Go Back to College
    <option value="s6">The Money Song
    <option value="s7">School for Monsters
    <option value="s8">The Money Song (Reprise)
    <option value="s9">There's a Fine, Fine Line (Reprise)
    <option value="s10">What Do You Do With a B.A. in English?
    (Reprise)
    <option value="s11">For Now
  </select>
</label>
```

#### 4.10.8 L' **data-list** élément



Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .

### **Contextes dans lesquels cet élément peut être utilisé :**

Où [le contenu du phrasé](#) est attendu.

### **Modèle de contenu :**

Soit : [phrasé contenu](#) .  
Ou : Zéro ou plus [option](#) et éléments [de support de script](#) .

### **Omission de balise dans text/html :**

Aucune des deux balises n'est omise.

### **Attributs de contenu :**

[Attributs globaux](#)

### **Considérations d'accessibilité :**

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

### **Interface DOM :**

[Exposed=Window]

```
interface HTMLDataListElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [SameObject] readonly attribute HTMLCollection options;
```

```
};
```

L' [datalist](#) élément représente un ensemble d' [option](#) éléments qui représentent des options prédéfinies pour d'autres contrôles. Dans le rendu, l' [datalist](#) élément ne [représente](#) rien et lui, ainsi que ses enfants, doivent être masqués.

L' [datalist](#) élément peut être utilisé de deux manières. Dans le cas le plus simple, l' [datalist](#) élément n'a que [option](#) des éléments enfants.

```
<label>
  Animal:
  <input name=animal list=animals>
  <datalist id=animals>
    <option value="Cat">
    <option value="Dog">
  </datalist>
</label>
```

Dans le cas le plus élaboré, l' [datalist](#) élément peut recevoir un contenu qui doit être affiché pour les clients de bas niveau qui ne prennent pas en charge [datalist](#). Dans ce cas, les [option](#) éléments sont prévus à l'intérieur d'un [select](#) élément à l'intérieur de l' [datalist](#) élément.

```
<label>
  Animal:
  <input name=animal list=animals>
</label>
<datalist id=animals>
  <label>
    or select from the list:
    <select name=animal>
      <option value="">
      <option>Cat
      <option>Dog
    </select>
  </label>
</datalist>
```

L' [datalist](#) élément est relié à un [input](#) élément à l'aide de l' [list](#) attribut sur l' [input](#) élément.

Chaque [option](#) élément qui est un descendant de l' [datalist](#) élément, qui n'est pas [disabled](#) et dont [la valeur](#) est une chaîne qui n'est pas la chaîne vide, représente une suggestion. Chaque suggestion a une [valeur](#) et une [étiquette](#) .

#### **[datalist.options](#)**

Renvoie un [HTMLCollection](#) des [option](#) éléments de l' [datalist](#) élément.

L' [options](#) attribut IDL doit renvoyer un [HTMLCollection](#) enraciné au [datalist](#) nœud, dont le filtre correspond [option](#) aux éléments.

**Validation de contrainte** : si un élément a un [datalist](#) ancêtre d'élément, il est [exclu de la validation de contrainte](#) .

#### **4.10.9 L' [optgroup](#) élément**



#### **[Catégories](#) :**

Aucun.

#### **[Contextes dans lesquels cet élément peut être utilisé](#) :**

En tant qu'enfant d'un [select](#) élément.

#### **[Modèle de contenu](#) :**

Zéro ou plus [option](#)et éléments [de support de script](#) .

### Omission de balise dans text/html :

[La balise de fin](#) d'un [optgroup](#)élément peut être omise si l' élément est immédiatement suivi d'un autre élément ou s'il n'y a plus de contenu dans l'élément parent.[optgroupoptgroup](#)

### Attributs de contenu :

[Attributs globaux](#)

[disabled](#)— Si le contrôle du formulaire est désactivé

[label](#)— Étiquette visible par l'utilisateur

### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### Interface DOM :

```
[Exposed=Window]

interface HTMLOptGroupElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute boolean disabled;

    [CEReactions] attribute DOMString label;

};
```

L' [optgroup](#)élément [représente](#) un groupe d' [option](#) éléments avec une étiquette commune.

Le groupe d'éléments de l'élément [option](#)est constitué des [option](#) éléments enfants de l' [optgroup](#)élément.

Lors de l'affichage [option](#)d'éléments dans [select](#)des éléments, les agents utilisateurs doivent montrer les [option](#)éléments de ces groupes comme étant liés les uns aux autres et séparés des autres [option](#)éléments.



L' [disabled](#)attribut est un [attribut booléen](#) et peut être utilisé pour [désactiver](#) un groupe d' [option](#)éléments ensemble.

L' [label](#) attribut doit être spécifié. Sa valeur donne le nom du groupe, pour les besoins de l'interface utilisateur. Les agents utilisateurs doivent utiliser la valeur de

cet attribut lors de l'étiquetage du groupe d' optionéléments dans un selectélément.

Les attributs **disabled** et **label** doivent refléter les attributs de contenu respectifs du même nom.

*Il n'y a aucun moyen de sélectionner un optgroupélément. Seuls optionles éléments peuvent être sélectionnés. Un optgroupélément fournit simplement une étiquette pour un groupe d' optionéléments.*

L'extrait suivant montre comment un ensemble de leçons de trois cours pourrait être proposé dans un selectwidget déroulant :

```
<form action="courseselector.dll" method="get">
  <p>Which course would you like to watch today?
  <p><label>Course:
    <select name="c">
      <optgroup label="8.01 Physics I: Classical Mechanics">
        <option value="8.01.1">Lecture 01: Powers of Ten
        <option value="8.01.2">Lecture 02: 1D Kinematics
        <option value="8.01.3">Lecture 03: Vectors
      <optgroup label="8.02 Electricity and Magnetism">
        <option value="8.02.1">Lecture 01: What holds our world
together?
        <option value="8.02.2">Lecture 02: Electric Field
        <option value="8.02.3">Lecture 03: Electric Flux
      <optgroup label="8.03 Physics III: Vibrations and Waves">
        <option value="8.03.1">Lecture 01: Periodic Phenomenon
        <option value="8.03.2">Lecture 02: Beats
        <option value="8.03.3">Lecture 03: Forced Oscillations with
Damping
    </select>
  </label>
  <p><input type="submit" value="► Play">
</form>
```

#### 4.10.10 L' optionélément



#### Catégories :

Aucun.

### Contextes dans lesquels cet élément peut être utilisé :

En tant qu'enfant d'un [select](#) élément.  
En tant qu'enfant d'un [datalist](#) élément.  
En tant qu'enfant d'un [optgroup](#) élément.

### Modèle de contenu :

Si l'élément a un [label](#) attribut et un [value](#) attribut : [Nothing](#) .  
Si l'élément a un [label](#) attribut mais pas [value](#) d'attribut : [Text](#) .  
Si l'élément n'a pas [label](#) d'attribut et n'est pas un enfant d'un [datalist](#) élément : [Texte](#) qui n'est pas [un espace inter-élément](#) .  
Si l'élément n'a pas [label](#) d'attribut et est un enfant d'un [datalist](#) élément : [Text](#) .

### Omission de balise dans text/html :

[La balise de fin](#) d'un [option](#) élément peut être omise si l'élément est immédiatement suivi d'un autre élément, ou s'il est immédiatement suivi d'un élément, ou s'il n'y a plus de contenu dans l'élément parent.[optionoptionoptgroup](#)

### Attributs de contenu :

#### Attributs globaux

[disabled](#)— Si le contrôle du formulaire est désactivé  
[label](#)— Étiquette visible par l'utilisateur  
[selected](#)— Si l'option est sélectionnée par défaut  
[value](#)— Valeur à utiliser pour [la soumission du formulaire](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

### Interface DOM :

```
[Exposed=Window,  
  
LegacyFactoryFunction=Option(optional DOMString text = "",  
optional DOMString value, optional boolean defaultSelected  
= false, optional boolean selected = false)]  
  
interface HTMLOptionElement : HTMLElement {  
  
    [HTMLConstructor] constructor();  
  
    [CEReactions] attribute boolean disabled;  
  
    readonly attribute HTMLFormElement? form;
```

```

[CEReactions] attribute DOMString label;

[CEReactions] attribute boolean defaultSelected;

attribute boolean selected;

[CEReactions] attribute DOMString value;

[CEReactions] attribute DOMString text;

readonly attribute long index;

};

```

L' option élément représente une option dans un select élément ou fait partie d'une liste de suggestions dans un datalist élément.

Dans certaines circonstances décrites dans la définition de l' select élément, un option élément peut être l' option d'étiquette d'espace réservé d'un élément . Une option d'étiquette d'espace réservé ne représente pas une option réelle, mais représente à la place une étiquette pour le contrôle. select



L' **disabled** attribut est un attribut booléen . Un option élément est **désactivé** si son disabled attribut est présent ou s'il est un enfant d'un optgroup élément dont disabled l'attribut est présent.

Un option élément désactivé doit empêcher tout événement mis en file d'attente sur la source de tâche d'interaction utilisateur d' être distribué sur l'élément. click

L' **label** attribut fournit une étiquette pour l'élément. L' **étiquette** d'un option élément est la valeur de l' label attribut content, s'il y en a un et que sa valeur n'est pas la chaîne vide, ou, sinon, la valeur de l' text attribut IDL de l'élément.

L' label attribut de contenu, s'il est spécifié, ne doit pas être vide.

L' **value** attribut fournit une valeur pour l'élément. La **valeur** d'un option élément est la valeur de l' value attribut content, s'il y en a un, ou, s'il n'y en a pas, la valeur de l' text attribut IDL de l'élément.

L' **selected** attribut est un attribut booléen . Il représente la sélection par défaut de l'élément.



La **saleté** d'un option élément est un état booléen, initialement faux. Il contrôle si l'ajout ou la suppression de l' selected attribut de contenu a un effet.

La **sélection** d'un option élément est un état booléen, initialement faux. Sauf indication contraire, lorsque l'élément est créé, sa sélection doit être définie sur true si l'élément a un selected attribut. Chaque fois que l'attribut option d'un élément selected est ajouté, si sa saleté est fausse, sa sélection doit être définie sur true. Chaque fois que l'attribut option d'un élément selected est supprimé, si sa saleté est fausse, sa sélection doit être définie sur faux.

*Le Option() constructeur, lorsqu'il est appelé avec trois arguments ou moins, remplace l'état initial de l' état de sélection pour qu'il soit toujours faux même si le troisième argument est vrai (ce qui implique qu'un selected attribut doit être défini). Le quatrième argument peut être utilisé pour définir explicitement l'état initial de sélection lors de l'utilisation du constructeur.*

Un select élément dont multiple l'attribut n'est pas spécifié ne doit pas avoir plus d'un option élément descendant avec son selected ensemble d'attributs.

L' **indice** d' un option élément est le nombre d' éléments qui sont dans la même liste d'options mais qui viennent avant lui dans l'ordre de l'arborescence . Si l' élément n'est pas dans une liste d'options , alors l' indice de l'élément est zéro.optionoptionoption

#### option.selected

Renvoie true si l'élément est sélectionné et false sinon.

Peut être défini pour remplacer l'état actuel de l'élément.

#### option.index

Renvoie l'index de l'élément dans sa liste select d'éléments options.

#### option.form

Renvoie l' form élément de l'élément, le cas échéant, ou null dans le cas contraire.

#### option.text

Identique à textContent, sauf que les espaces sont réduits et script que les éléments sont ignorés.

```
option = new Option([ text [, value [, defaultSelected [, selected ] ] ] ] )
```

✓

Renvoie un nouvel option élément.

L' argument *text* définit le contenu de l'élément.

L' argument *valeur* définit l' value attribut.

L' argument *defaultSelected* définit l' selected attribut.

L' argument *sélectionné* définit si l'élément est sélectionné ou non. S'il est omis, même si l' argument *defaultSelected* est vrai, l'élément n'est pas sélectionné.

L' **disabled**attribut IDL doit [réfléter](#) l'attribut de contenu du même nom. L' **defaultSelected**attribut IDL doit [réfléter](#) l' [selected](#)attribut content.

L' **label** attribut IDL, lors de l'obtention, s'il existe un [label](#)attribut de contenu, doit renvoyer la valeur de cet attribut ; sinon, il doit renvoyer le [label](#) de l'élément . Lors de la définition, l' [label](#)attribut de contenu de l'élément doit être défini sur la nouvelle valeur.

L' **value** attribut IDL, lors de l'obtention, doit renvoyer la [valeur](#) de l'élément . Lors de la définition, l' [value](#)attribut de contenu de l'élément doit être défini sur la nouvelle valeur.

L' **selected**attribut IDL, à l'obtention, doit retourner true si la [sélection](#) de l'élément est true, et false sinon. Lors de la définition, il doit définir la [sélection](#) de l'élément sur la nouvelle valeur, définir sa [sauté](#) sur true, puis obliger l'élément à [demander une réinitialisation](#) .

L' **index** attribut IDL doit retourner l' [index](#) de l'élément .

L' **text**attribut IDL, lors de l'obtention, doit renvoyer le résultat de [la suppression et de la réduction des espaces blancs ASCII](#) à partir de la concaténation des [données](#) de tous les [Text](#)nœuds descendants de l' [option](#)élément, dans [l'ordre de l'arborescence](#) , à l'exclusion de ceux qui sont des descendants de descendants de l' [option](#)élément qui sont eux-mêmes [script](#)ou [SVGscript](#) éléments.

Le [text](#)setter de l'attribut doit [chaîne remplacer all](#) par la valeur donnée dans cet élément.

Le **form**comportement de l'attribut IDL dépend de si l' [option](#)élément est dans un [select](#)élément ou non. Si le [option](#)a un [select](#)élément comme parent, ou a un [optgroup](#)élément comme parent et que cet [optgroup](#)élément a un [select](#)élément comme parent, alors l' **form**attribut IDL doit renvoyer la même valeur que l' **form**attribut IDL sur cet [select](#)élément. Sinon, il doit retourner null.

Une fonction de fabrique héritée est fournie pour créer [HTMLOptionElement](#)des objets (en plus des méthodes de fabrique du DOM telles que [createElement\(\)](#)): . Lorsqu'elle est invoquée, la fonction de fabrique héritée doit effectuer les étapes suivantes :[Option\(text, value, defaultSelected, selected\)](#)

1. Soit *document* l' objet [global actuel associé](#)[Document](#) .

2. Soit *option* le résultat de [la création d'un élément](#) donné *document* , [option](#) et l' [espace de noms HTML](#) .
3. Si *text* n'est pas la chaîne vide, ajoutez à l'*option* un nouveau [Text](#) nœud dont les données sont *text* .
4. Si *value* est donné, alors [définissez une valeur d'attribut](#) pour *option* en utilisant " [value](#) " et *value* .
5. Si *defaultSelected* est vrai, [définissez une valeur d'attribut](#) pour l'*option* en utilisant " [selected](#) " et la chaîne vide.
6. Si *sélectionné* est vrai, alors définissez [la sélection](#) de l' *option* sur vrai ; sinon, définissez sa [sélection](#) sur false (même si *defaultSelected* est true).
7. Possibilité de retour .

#### 4.10.11 L' [textarea](#) élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu interactif](#) .  
[Listed](#) , [labelable](#) , [submittable](#) , [resettable](#) et [autocapitalize-heriting form-associated element](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu du phrasé](#) est attendu.

##### Modèle de contenu :

[Texte](#) .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
[autocomplete](#)- Astuce pour la fonction de remplissage automatique du formulaire  
[cols](#)— Nombre maximum de caractères par ligne  
[dirname](#)— Nom du contrôle de formulaire à utiliser pour envoyer la [directionnalité](#) de l'élément lors de [la soumission du formulaire](#)  
[disabled](#)— Si le contrôle du formulaire est désactivé  
[form](#)— Associe l'élément à un [form](#) élément

maxlength— [Longueur](#) maximale de la valeur  
minlength— [Longueur](#) minimale de la valeur  
name— Nom de l'élément à utiliser pour [la soumission du formulaire](#) et dans l' `form.elements` API  
placeholder— Étiquette visible par l'utilisateur à placer dans le contrôle du formulaire  
readonly— Autoriser ou non la modification de la valeur par l'utilisateur  
required— Si le contrôle est requis pour [la soumission du formulaire](#)  
rows— Nombre de lignes à afficher  
wrap— Comment la valeur du contrôle de formulaire doit être enveloppée pour [la soumission du formulaire](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .

[Pour les exécutants](#) .

### Interface DOM :

[Exposed=Window]

interface **HTMLTextAreaElement** : [HTMLElement](#) {

[[HTMLConstructor](#)] constructor();

[[CEReactions](#)] attribute DOMString [autocomplete](#);

[[CEReactions](#)] attribute unsigned long [cols](#);

[[CEReactions](#)] attribute DOMString [dirName](#);

[[CEReactions](#)] attribute boolean [disabled](#);

readonly attribute [HTMLFormElement](#)? [form](#);

[[CEReactions](#)] attribute long [maxLength](#);

[[CEReactions](#)] attribute long [minLength](#);

[[CEReactions](#)] attribute DOMString [name](#);

[[CEReactions](#)] attribute DOMString [placeholder](#);

[[CEReactions](#)] attribute boolean [readOnly](#);

[[CEReactions](#)] attribute boolean [required](#);

[[CEReactions](#)] attribute unsigned long [rows](#);

[[CEReactions](#)] attribute DOMString [wrap](#);

readonly attribute DOMString [type](#);

[[CEReactions](#)] attribute DOMString [defaultValue](#);

attribute [[LegacyNullToEmptyString](#)] DOMString [value](#);

readonly attribute unsigned long [textLength](#);

readonly attribute boolean [willValidate](#);

readonly attribute [ValidityState](#) [validity](#);

readonly attribute DOMString [validationMessage](#);

boolean [checkValidity](#)();

boolean [reportValidity](#)();

undefined [setCustomValidity](#)(DOMString error);

readonly attribute [NodeList](#) [labels](#);

undefined [select](#)();

attribute unsigned long [selectionStart](#);

attribute unsigned long [selectionEnd](#);

attribute DOMString [selectionDirection](#);

undefined [setRangeText](#)(DOMString replacement);

undefined [setRangeText](#)(DOMString replacement, unsigned

long start, unsigned long end, optional [SelectionMode](#)

selectionMode = "preserve");

```
undefined setSelectionRange(unsigned long start, unsigned  
long end, optional DOMString direction);  
};
```

The [textarea](#) element [represents](#) a multiline plain text edit control for the element's **raw value**. The contents of the control represent the control's default value.

The [raw value](#) of a [textarea](#) control must be initially the empty string.

*This element [has rendering requirements involving the bidirectional algorithm](#).*

The **readonly** attribute is a [boolean attribute](#) used to control whether the text can be edited by the user or not.

In this example, a text control is marked read-only because it represents a read-only file:

```
Filename: <code>/etc/bash.bashrc</code>  
<textarea name="buffer" readonly>  
# System-wide .bashrc file for interactive bash(1) shells.  
  
# To enable the settings / commands in this file for login shells  
as well,  
# this file has to be sourced in /etc/profile.  
  
# If not running interactively, don't do anything  
[ -z "$PS1" ] && return  
...</textarea>
```

**Constraint validation:** If the [readonly](#) attribute is specified on a [textarea](#) element, the element is [barred from constraint validation](#).

A [textarea](#) element is [mutable](#) if it is neither [disabled](#) nor has a [readonly](#) attribute specified.

When a [textarea](#) is [mutable](#), its [raw value](#) should be editable by the user: the user agent should allow the user to edit, insert, and remove text, and to insert and remove line breaks in the form of U+000A LINE FEED (LF) characters. Any time the user causes the element's [raw value](#) to change, the user agent must [queue an element task](#) on the [user interaction task source](#) given the [textarea](#) element to [fire an event](#) named [input](#) at the [textarea](#) element, with the [bubbles](#) and [composed](#) attributes initialized to true. User agents may wait for a

suitable break in the user's interaction before queuing the task; for example, a user agent could wait for the user to have not hit a key for 100ms, so as to only fire the event when the user pauses, instead of continuously for each keystroke.

A [textarea](#) element's [dirty value flag](#) must be set to true whenever the user interacts with the control in a way that changes the [raw value](#).

The [cloning steps](#) for [textarea](#) elements must propagate the [raw value](#) and [dirty value flag](#) from the node being cloned to the copy.

The [children changed steps](#) for [textarea](#) elements must, if the element's [dirty value flag](#) is false, set the element's [raw value](#) to its [child text content](#).

The [reset algorithm](#) for [textarea](#) elements is to set the [dirty value flag](#) back to false, and set the [raw value](#) of element to its [child text content](#).

When a [textarea](#) element is popped off the [stack of open elements](#) of an [HTML parser](#) or [XML parser](#), then the user agent must invoke the element's [reset algorithm](#).

If the element is [mutable](#), the user agent should allow the user to change the writing direction of the element, setting it either to a left-to-right writing direction or a right-to-left writing direction. If the user does so, the user agent must then run the following steps:

1. Set the element's [dir](#) attribute to "[ltr](#)" if the user selected a left-to-right writing direction, and "[rtl](#)" if the user selected a right-to-left writing direction.
2. [Queue an element task](#) on the [user interaction task source](#) given the [textarea](#) element to [fire an event](#) named [input](#) at the [textarea](#) element, with the [bubbles](#) and [composed](#) attributes initialized to true.

The [cols](#) attribute specifies the expected maximum number of characters per line. If the [cols](#) attribute is specified, its value must be a [valid non-negative integer](#) greater than zero. If applying the [rules for parsing non-negative integers](#) to the attribute's value results in a number greater than zero, then the element's **character width** is that value; otherwise, it is 20.

The user agent may use the [textarea](#) element's [character width](#) as a hint to the user as to how many characters the server prefers per line (e.g. for visual user agents by making the width of the control be that many characters). In visual renderings, the user agent should wrap the user's input in the rendering so that each line is no wider than this number of characters.

The [rows](#) attribute specifies the number of lines to show. If the [rows](#) attribute is specified, its value must be a [valid non-negative integer](#) greater than zero. If applying the [rules for parsing non-negative integers](#) to the attribute's value results in a number greater than zero, then the element's **character height** is that value; otherwise, it is 2.

Visual user agents should set the height of the control to the number of lines given by [character height](#).

The **wrap** attribute is an [enumerated attribute](#) with two keywords and states: the **soft** keyword which maps to the [Soft](#) state, and the **hard** keyword which maps to the [Hard](#) state. The [missing value default](#) and [invalid value default](#) are the [Soft](#) state.

The **Soft** state indicates that the text in the [textarea](#) is not to be wrapped when it is submitted (though it can still be wrapped in the rendering).

The **Hard** state indicates that the text in the [textarea](#) is to have newlines added by the user agent so that the text is wrapped when it is submitted.

If the element's **wrap** attribute is in the [Hard](#) state, the [cols](#) attribute must be specified.

For historical reasons, the element's value is normalized in three different ways for three different purposes. The [raw value](#) is the value as it was originally set. It is not normalized. The [API value](#) is the value used in the [value](#) IDL attribute, [textLength](#) IDL attribute, and by the [maxlength](#) and [minlength](#) content attributes. It is normalized so that line breaks use U+000A LINE FEED (LF) characters. Finally, there is the [value](#), as used in form submission and other processing models in this specification. It is normalized as for the [API value](#), and in addition, if necessary given the element's **wrap** attribute, additional line breaks are inserted to wrap the text at the given width.

The algorithm for obtaining the element's [API value](#) is to return the element's [raw value](#), with [newlines normalized](#).

The element's [value](#) is defined to be the element's [API value](#) with the [textarea wrapping transformation](#) applied. The **textarea wrapping transformation** is the following algorithm, as applied to a string:

1. If the element's **wrap** attribute is in the [Hard](#) state, insert U+000A LINE FEED (LF) characters into the string using an [implementation-defined](#) algorithm so that each line has no more than [character width](#) characters. For the purposes of this requirement, lines are delimited by the start of the string, the end of the string, and U+000A LINE FEED (LF) characters.

The **maxlength** attribute is a [form control maxlength attribute](#).

If the [textarea](#) element has a [maximum allowed value length](#), then the element's children must be such that the [length](#) of the value of the element's [descendant text content](#) with [newlines normalized](#) is equal to or less than the element's [maximum allowed value length](#).

The **minlength** attribute is a [form control minlength attribute](#).



The **required** attribute is a [boolean attribute](#). When specified, the user will be required to enter a value before submitting the form.

**Constraint validation:** If the element has its [required](#) attribute specified, and the element is [mutable](#), and the element's [value](#) is the empty string, then the element is [suffering from being missing](#).

The **placeholder** attribute represents a *short* hint (a word or short phrase) intended to aid the user with data entry when the control has no value. A hint could be a sample value or a brief description of the expected format.

The [placeholder](#) attribute should not be used as an alternative to a [label](#). For a longer hint or other advisory text, the [title](#) attribute is more appropriate.

*These mechanisms are very similar but subtly different: the hint given by the control's [label](#) is shown at all times; the short hint given in the [placeholder](#) attribute is shown before the user enters a value; and the hint in the [title](#) attribute is shown when the user requests further help.*

User agents should present this hint to the user when the element's [value](#) is the empty string and the control is not [focused](#) (e.g. by displaying it inside a blank unfocused control). All U+000D CARRIAGE RETURN U+000A LINE FEED character pairs (CRLF) in the hint, as well as all other U+000D CARRIAGE RETURN (CR) and U+000A LINE FEED (LF) characters in the hint, must be treated as line breaks when rendering the hint.

If a user agent normally doesn't show this hint to the user when the control is [focused](#), then the user agent should nonetheless show the hint for the control if it was focused as a result of the [autofocus](#) attribute, since in that case the user will not have had an opportunity to examine the control before focusing it.

The [name](#) attribute represents the element's name. The [dirname](#) attribute controls how the element's [directionality](#) is submitted. The [disabled](#) attribute is used to make the control non-interactive and to prevent its value from being submitted. The [form](#) attribute is used to explicitly associate the [textarea](#) element with its [form owner](#). The [autocomplete](#) attribute controls how the user agent provides autofill behavior.

#### **textarea.type**

Returns the string "textarea".

#### **textarea.value**

Returns the current value of the element.

Can be set, to change the value.

The **cols**, **placeholder**, **required**, **rows**, and **wrap** IDL attributes must [reflect](#) the respective content attributes of the same name. The **cols** and **rows** attributes are [limited to only non-negative numbers greater than zero with fallback](#).

The `cols` IDL attribute's default value is 20. The `rows` IDL attribute's default value is 2. The `dirName` IDL attribute must [reflect](#) the `dirname` content attribute. The `maxLength` IDL attribute must [reflect](#) the `maxlength` content attribute, [limited to only non-negative numbers](#). The `minLength` IDL attribute must [reflect](#) the `minlength` content attribute, [limited to only non-negative numbers](#). The `readOnly` IDL attribute must [reflect](#) the `readonly` content attribute.

The `type` IDL attribute must return the value `"textarea"`.

The `defaultValue` attribute's getter must return the element's [child text content](#).

The `defaultValue` attribute's setter must [string replace all](#) with the given value within this element.

The `value` IDL attribute must, on getting, return the element's [API value](#). On setting, it must perform the following steps:

1. Let *oldAPIValue* be this element's [API value](#).
2. Set this element's [raw value](#) to the new value.
3. Set this element's [dirty value flag](#) to true.
4. If the new [API value](#) is different from *oldAPIValue*, then move the [text entry cursor position](#) to the end of the text control, unselecting any selected text and [resetting the selection direction](#) to `"none"`.

The `textLength` IDL attribute must return the [length](#) of the element's [API value](#).

The `willValidate`, `validity`, and `validationMessage` IDL attributes, and the `checkValidity()`, `reportValidity()`, and `setCustomValidity()` methods, are part of the [constraint validation API](#).

The `labels` IDL attribute provides a list of the element's [labels](#).

The `select()`, `selectionStart`, `selectionEnd`, `selectionDirection`, `setRangeText()`, and `setSelectionRange()` methods and IDL attributes expose the element's text selection. The `disabled`, `form`, and `name` IDL attributes are part of the element's forms API.

Here is an example of a `textarea` being used for unrestricted free-form text input in a form:

```
<p>If you have any comments, please let us know: <textarea cols=80  
name=comments></textarea></p>
```

To specify a maximum length for the comments, one can use the `maxlength` attribute:

```
<p>If you have any short comments, please let us know: <textarea  
cols=80 name=comments maxlength=200></textarea></p>
```

To give a default value, text can be included inside the element:

```
<p>If you have any comments, please let us know: <textarea cols=80  
name=comments>You rock!</textarea></p>
```

You can also give a minimum length. Here, a letter needs to be filled out by the user; a template (which is shorter than the minimum length) is provided, but is insufficient to submit the form:

```
<textarea required minlength="500">Dear Madam Speaker,  
  
Regarding your letter dated ...  
  
...  
  
Yours Sincerely,  
  
...</textarea>
```

A placeholder can be given as well, to suggest the basic form to the user, without providing an explicit template:

```
<textarea placeholder="Dear Francine,  
  
They closed the parks this week, so we won't be able to  
meet you there. Should we just have dinner?  
  
Love,  
Daddy"></textarea>
```

To have the browser submit [the directionality](#) of the element along with the value, the [dirname](#) attribute can be specified:

```
<p>If you have any comments, please let us know (you may use either  
English or Hebrew for your comments):  
  
<textarea cols=80 name=comments  
dirname=comments.dir></textarea></p>
```

#### 4.10.12 The **output** element



**Categories:**

[Flow content](#).

[Phrasing content](#).

[Listed](#), [labelable](#), [resettable](#), and [autocapitalize-inheriting form-associated element](#).

[Palpable content](#).

### **Contexts in which this element can be used:**

Where [phrasing content](#) is expected.

### **Content model:**

[Phrasing content](#).

### **Tag omission in text/html:**

Neither tag is omissible.

### **Content attributes:**

[Global attributes](#)

[for](#) — Specifies controls from which the output was calculated

[form](#) — Associates the element with a [form](#) element

[name](#) — Name of the element to use in the [form.elements](#) API.

### **Accessibility considerations:**

[For authors](#).

[For implementers](#).

### **DOM interface:**

[Exposed=Window]

```
interface HTMLInputElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [SameObject, PutForwards=value] readonly attribute
```

```
    DOMTokenList htmlFor;
```

```
  readonly attribute HTMLFormElement? form;
```

```
  [CEReactions] attribute DOMString name;
```

```
  readonly attribute DOMString type;
```

```
  [CEReactions] attribute DOMString defaultValue;
```

```
  [CEReactions] attribute DOMString value;
```

```

readonly attribute boolean willValidate;

readonly attribute ValidityState validity;

readonly attribute DOMString validationMessage;

boolean checkValidity();

boolean reportValidity();

undefined setCustomValidity(DOMString error);

readonly attribute NodeList labels;

};

```

The [output](#) element [represents](#) the result of a calculation performed by the application, or the result of a user action.

*This element can be contrasted with the [samp](#) element, which is the appropriate element for quoting the output of other programs run previously.*

✓MDN

The [for](#) content attribute allows an explicit relationship to be made between the result of a calculation and the elements that represent the values that went into the calculation or that otherwise influenced the calculation. The [for](#) attribute, if specified, must contain a string consisting of an [unordered set of unique space-separated tokens](#), none of which are [identical to](#) another token and each of which must have the value of an [ID](#) of an element in the same [tree](#).

The [form](#) attribute is used to explicitly associate the [output](#) element with its [form owner](#). The [name](#) attribute represents the element's name. The [output](#) element is associated with a form so that it can be easily [referenced](#) from the event handlers of form controls; the element's value itself is not submitted when the form is submitted.

The element has a **default value override** (null or a string). Initially it must be null.

The element's **default value** is determined by the following steps:

1. If this element's [default value override](#) is non-null, then return it.
2. Return this element's [descendant text content](#).

The [reset algorithm](#) for [output](#) elements is to run these steps:

1. [String replace all](#) with this element's [default value](#) within this element.
2. Set this element's [default value override](#) to null.

**`output.value`** [ = *value* ]

Returns the element's current value.

Can be set, to change the value.

**`output.defaultValue`** [ = *value* ]

Returns the element's current default value.

Can be set, to change the default value.

**`output.type`**

Returns the string "output".

The **value** getter steps are to return [this](#)'s [descendant text content](#).

The **value** setter steps are:

1. Set [this](#)'s [default value override](#) to its [default value](#).
2. [String replace all](#) with the given value within [this](#).

The **defaultValue** getter steps are to return the result of running [this](#)'s [default value](#).

The **defaultValue** setter steps are:

1. If [this](#)'s [default value override](#) is null, then [string replace all](#) with the given value within [this](#) and return.
2. Set [this](#)'s [default value override](#) to the given value.

The **type** getter steps are to return "output".

The **htmlFor** IDL attribute must [reflect](#) the [for](#) content attribute.

The [willValidate](#), [validity](#), and [validationMessage](#) IDL attributes, and the [checkValidity\(\)](#), [reportValidity\(\)](#), and [setCustomValidity\(\)](#) methods, are part of the [constraint validation API](#). The [labels](#) IDL attribute provides a list of the element's [labels](#). The [form](#) and [name](#) IDL attributes are part of the element's forms API.

A simple calculator could use [output](#) for its display of calculated results:

```
<form onsubmit="return false" oninput="o.value = a.valueAsNumber +
b.valueAsNumber">
  <input id=a type=number step=any> +
  <input id=b type=number step=any> =
```

```
<output id=o for="a b"></output>
</form>
```

In this example, an [output](#) element is used to report the results of a calculation performed by a remote server, as they come in:

```
<output id="result"></output>
<script>
  var primeSource = new WebSocket('ws://primes.example.net/');
  primeSource.onmessage = function (event) {
    document.getElementById('result').value = event.data;
  }
</script>
```

#### 4.10.13 The [progress](#) element



##### Categories:

[Flow content](#).  
[Phrasing content](#).  
[Labelable element](#).  
[Palpable content](#).

##### Contexts in which this element can be used:

Where [phrasing content](#) is expected.

##### Content model:

[Phrasing content](#), but there must be no [progress](#) element descendants.

##### Tag omission in text/html:

Neither tag is omissible.

##### Content attributes:

[Global attributes](#)  
[value](#) — Current value of the element  
[max](#) — Upper bound of range

##### Accessibility considerations:

[For authors](#).  
[For implementers](#).

##### DOM interface:

[Exposed=Window]
------------------

```

interface HTMLProgressElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute double value;

    [CEReactions] attribute double max;

    readonly attribute double position;

    readonly attribute NodeList labels;

};

```

The [progress](#) element [represents](#) the completion progress of a task. The progress is either indeterminate, indicating that progress is being made but that it is not clear how much more work remains to be done before the task is complete (e.g. because the task is waiting for a remote host to respond), or the progress is a number in the range zero to a maximum, giving the fraction of work that has so far been completed.



There are two attributes that determine the current task completion represented by the element. The [value](#) attribute specifies how much of the task has been completed, and the [max](#) attribute specifies how much work the task requires in total. The units are arbitrary and not specified.

*To make a determinate progress bar, add a [value](#) attribute with the current progress (either a number from 0.0 to 1.0, or, if the [max](#) attribute is specified, a number from 0 to the value of the [max](#) attribute). To make an indeterminate progress bar, remove the [value](#) attribute.*

Authors are encouraged to also include the current value and the maximum value inline as text inside the element, so that the progress is made available to users of legacy user agents.

Here is a snippet of a web application that shows the progress of some automated task:

```

<section>
  <h2>Task Progress</h2>
  <p>Progress: <progress id=p max=100><span>0</span>%</progress></p>
  <script>
    var progressBar = document.getElementById('p');

```



```
function updateProgress(newValue) {
    progressBar.value = newValue;
    progressBar.getElementsByTagName('span')[0].textContent =
newValue;
}
</script>
</section>
```

(The `updateProgress()` method in this example would be called by some other code on the page to update the actual progress bar as the task progressed.)

The `value` and `max` attributes, when present, must have values that are [valid floating-point numbers](#). The `value` attribute, if present, must have a value equal to or greater than zero, and less than or equal to the value of the `max` attribute, if present, or 1.0, otherwise. The `max` attribute, if present, must have a value greater than zero.

*The `progress` element is the wrong element to use for something that is just a gauge, as opposed to task progress. For instance, indicating disk space usage using `progress` would be inappropriate. Instead, the `meter` element is available for such use cases.*

**User agent requirements:** If the `value` attribute is omitted, then the progress bar is an indeterminate progress bar. Otherwise, it is a determinate progress bar.

If the progress bar is a determinate progress bar and the element has a `max` attribute, the user agent must parse the `max` attribute's value according to the [rules for parsing floating-point number values](#). If this does not result in an error, and if the parsed value is greater than zero, then the **maximum value** of the progress bar is that value. Otherwise, if the element has no `max` attribute, or if it has one but parsing it resulted in an error, or if the parsed value was less than or equal to zero, then the [maximum value](#) of the progress bar is 1.0.

If the progress bar is a determinate progress bar, user agents must parse the `value` attribute's value according to the [rules for parsing floating-point number values](#). If this does not result in an error and the parsed value is greater than zero, then the **value** of the progress bar is that parsed value. Otherwise, if parsing the `value` attribute's value resulted in an error or a number less than or equal to zero, then the [value](#) of the progress bar is zero.

If the progress bar is a determinate progress bar, then the **current value** is the [maximum value](#), if [value](#) is greater than the [maximum value](#), and [value](#) otherwise.

**UA requirements for showing the progress bar:** When representing a `progress` element to the user, the UA should indicate whether it is a determinate or indeterminate progress bar, and in the former case, should indicate the relative position of the [current value](#) relative to the [maximum value](#).

**`progress.position`**

For a determinate progress bar (one with known current and maximum values), returns the result of dividing the current value by the maximum value.  
For an indeterminate progress bar, returns `-1`.

If the progress bar is an indeterminate progress bar, then the `position` IDL attribute must return `-1`. Otherwise, it must return the result of dividing the [current value](#) by the [maximum value](#).

If the progress bar is an indeterminate progress bar, then the `value` IDL attribute, on getting, must return `0`. Otherwise, it must return the [current value](#). On setting, the given value must be converted to the [best representation of the number as a floating-point number](#) and then the `value` content attribute must be set to that string.

*Setting the `value` IDL attribute to itself when the corresponding content attribute is absent would change the progress bar from an indeterminate progress bar to a determinate progress bar with no progress.*

The `max` IDL attribute must [reflect](#) the content attribute of the same name, [limited to numbers greater than zero](#). The default value for `max` is `1.0`.

The `labels` IDL attribute provides a list of the element's [labels](#).

#### 4.10.14 The `meter` element



##### Categories:

[Flow content](#).  
[Phrasing content](#).  
[Labelable element](#).  
[Palpable content](#).

##### Contexts in which this element can be used:

Where [phrasing content](#) is expected.

##### Content model:

[Phrasing content](#), but there must be no `meter` element descendants.

##### Tag omission in text/html:

Neither tag is omissible.

##### Content attributes:

[Global attributes](#)  
`value` — Current value of the element  
`min` — Lower bound of range

max — Upper bound of range  
low — High limit of low range  
high — Low limit of high range  
optimum — Optimum value in gauge

### Accessibility considerations:

For authors.

For implementers.

### DOM interface:

```
[Exposed=Window]

interface HTMLMeterElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute double value;

    [CEReactions] attribute double min;

    [CEReactions] attribute double max;

    [CEReactions] attribute double low;

    [CEReactions] attribute double high;

    [CEReactions] attribute double optimum;

    readonly attribute NodeList labels;

};
```

The meter element represents a scalar measurement within a known range, or a fractional value; for example disk usage, the relevance of a query result, or the fraction of a voting population to have selected a particular candidate.

This is also known as a gauge.

The meter element should not be used to indicate progress (as in a progress bar). For that role, HTML provides a separate progress element.

*The meter element also does not represent a scalar value of arbitrary range — for example, it would be wrong to use this to report a weight, or height, unless there is a known maximum value.*

There are six attributes that determine the semantics of the gauge represented by the element.



The **min** attribute specifies the lower bound of the range, and the **max** attribute specifies the upper bound. The **value** attribute specifies the value to have the gauge indicate as the "measured" value.

The other three attributes can be used to segment the gauge's range into "low", "medium", and "high" parts, and to indicate which part of the gauge is the "optimum" part. The **low** attribute specifies the range that is considered to be the "low" part, and the **high** attribute specifies the range that is considered to be the "high" part. The **optimum** attribute gives the position that is "optimum"; if that is higher than the "high" value then this indicates that the higher the value, the better; if it's lower than the "low" mark then it indicates that lower values are better, and naturally if it is in between then it indicates that neither high nor low values are good.

**Authoring requirements:** The **value** attribute must be specified.

The **value**, **min**, **low**, **high**, **max**, and **optimum** attributes, when present, must have values that are [valid floating-point numbers](#).

In addition, the attributes' values are further constrained:

Let *value* be the **value** attribute's number.

If the **min** attribute is specified, then let *minimum* be that attribute's value; otherwise, let it be zero.

If the **max** attribute is specified, then let *maximum* be that attribute's value; otherwise, let it be 1.0.

The following inequalities must hold, as applicable:

- $minimum \leq value \leq maximum$
- $minimum \leq low \leq maximum$  (if **low** is specified)
- $minimum \leq high \leq maximum$  (if **high** is specified)
- $minimum \leq optimum \leq maximum$  (if **optimum** is specified)
- $low \leq high$  (if both **low** and **high** are specified)

*If no minimum or maximum is specified, then the range is assumed to be 0..1, and the value thus has to be within that range.*

Authors are encouraged to include a textual representation of the gauge's state in the element's contents, for users of user agents that do not support the **meter** element.

When used with [microdata](#), the [meter](#) element's [value](#) attribute provides the element's machine-readable value.

The following examples show three gauges that would all be three-quarters full:

```
Storage space usage: <meter value=6 max=8>6 blocks used (out of 8
total)</meter>
Voter turnout: <meter value=0.75></meter>
Tickets sold: <meter min="0" max="100" value="75"></meter>
```

The following example is incorrect use of the element, because it doesn't give a range (and since the default maximum is 1, both of the gauges would end up looking maxed out):

```
<p>The grapefruit pie had a radius of <meter value=12>12cm</meter>
and a height of <meter value=2>2cm</meter>.</p> <!-- BAD! -->
```

Instead, one would either not include the meter element, or use the meter element with a defined range to give the dimensions in context compared to other pies:

```
<p>The grapefruit pie had a radius of 12cm and a height of
2cm.</p>
<dl>
  <dt>Radius: <dd> <meter min=0 max=20 value=12>12cm</meter>
  <dt>Height: <dd> <meter min=0 max=10 value=2>2cm</meter>
</dl>
```

There is no explicit way to specify units in the [meter](#) element, but the units may be specified in the [title](#) attribute in free-form text.

The example above could be extended to mention the units:

```
<dl>
  <dt>Radius: <dd> <meter min=0 max=20 value=12
title="centimeters">12cm</meter>
  <dt>Height: <dd> <meter min=0 max=10 value=2
title="centimeters">2cm</meter>
</dl>
```

**User agent requirements:** User agents must parse the [min](#), [max](#), [value](#), [low](#), [high](#), and [optimum](#) attributes using the [rules for parsing floating-point number values](#).

User agents must then use all these numbers to obtain values for six points on the gauge, as follows. (The order in which these are evaluated is important, as some of the values refer to earlier ones.)

### **The *minimum value***

If the min attribute is specified and a value could be parsed out of it, then the minimum value is that value. Otherwise, the minimum value is zero.

### **The *maximum value***

If the max attribute is specified and a value could be parsed out of it, then the candidate maximum value is that value. Otherwise, the candidate maximum value is 1.0.

If the candidate maximum value is greater than or equal to the minimum value, then the maximum value is the candidate maximum value. Otherwise, the maximum value is the same as the minimum value.

### **The *actual value***

If the value attribute is specified and a value could be parsed out of it, then that value is the candidate actual value. Otherwise, the candidate actual value is zero.

If the candidate actual value is less than the minimum value, then the actual value is the minimum value.

Otherwise, if the candidate actual value is greater than the maximum value, then the actual value is the maximum value.

Otherwise, the actual value is the candidate actual value.

### **The *low boundary***

If the low attribute is specified and a value could be parsed out of it, then the candidate low boundary is that value. Otherwise, the candidate low boundary is the same as the minimum value.

If the candidate low boundary is less than the minimum value, then the low boundary is the minimum value.

Otherwise, if the candidate low boundary is greater than the maximum value, then the low boundary is the maximum value.

Otherwise, the low boundary is the candidate low boundary.

### **The *high boundary***

If the high attribute is specified and a value could be parsed out of it, then the candidate high boundary is that value. Otherwise, the candidate high boundary is the same as the maximum value.

If the candidate high boundary is less than the low boundary, then the high boundary is the low boundary.

Otherwise, if the candidate high boundary is greater than the maximum value, then the high boundary is the maximum value.

Otherwise, the high boundary is the candidate high boundary.

### The *optimum point*

If the optimum attribute is specified and a value could be parsed out of it, then the candidate optimum point is that value. Otherwise, the candidate optimum point is the midpoint between the minimum value and the maximum value.

If the candidate optimum point is less than the minimum value, then the optimum point is the minimum value.

Otherwise, if the candidate optimum point is greater than the maximum value, then the optimum point is the maximum value.

Otherwise, the optimum point is the candidate optimum point.

All of which will result in the following inequalities all being true:

- minimum value  $\leq$  actual value  $\leq$  maximum value
- minimum value  $\leq$  low boundary  $\leq$  high boundary  $\leq$  maximum value
- minimum value  $\leq$  optimum point  $\leq$  maximum value

**UA requirements for regions of the gauge:** If the optimum point is equal to the low boundary or the high boundary, or anywhere in between them, then the region between the low and high boundaries of the gauge must be treated as the optimum region, and the low and high parts, if any, must be treated as suboptimal. Otherwise, if the optimum point is less than the low boundary, then the region between the minimum value and the low boundary must be treated as the optimum region, the region from the low boundary up to the high boundary must be treated as a suboptimal region, and the remaining region must be treated as an even less good region. Finally, if the optimum point is higher than the high boundary, then the situation is reversed; the region between the high boundary and the maximum value must be treated as the optimum region, the region from the high boundary down to the low boundary must be treated as a suboptimal region, and the remaining region must be treated as an even less good region.

**UA requirements for showing the gauge:** When representing a meter element to the user, the UA should indicate the relative position of the actual value to the minimum and maximum values, and the relationship between the actual value and the three regions of the gauge.

The following markup:

```
<h3>Suggested groups</h3>
<menu>
  <li><a href="?cmd=hsg" onclick="hideSuggestedGroups()">Hide
suggested groups</a></li>
```


```


</menu>
<ul>
  <li>
    <p><a
href="/group/comp.infosystems.www.authoring.stylesheets/view">comp.
infosystems.www.authoring.stylesheets</a> -
      <a
href="/group/comp.infosystems.www.authoring.stylesheets/subscribe">
join</a></p>
    <p>Group description: <strong>Layout/presentation on the
WWW.</strong></p>
    <p><meter value="0.5">Moderate activity,</meter> Usenet, 618
subscribers</p>
  </li>
  <li>
    <p><a
href="/group/netcape.public.mozilla.xpinstall/view">netcape.publi
c.mozilla.xpinstall</a> -
      <a
href="/group/netcape.public.mozilla.xpinstall/subscribe">join</a><
/p>
    <p>Group description: <strong>Mozilla XPInstall
discussion.</strong></p>
    <p><meter value="0.25">Low activity,</meter> Usenet, 22
subscribers</p>
  </li>
  <li>
    <p><a
href="/group/mozilla.dev.general/view">mozilla.dev.general</a> -
      <a href="/group/mozilla.dev.general/subscribe">join</a></p>
    <p><meter value="0.25">Low activity,</meter> Usenet, 66
subscribers</p>
  </li>
</ul>


```

Might be rendered as follows:

**Suggested groups** - [Hide suggested groups](#)

[comp.infosystems.www.authoring.stylesheets](#) - [join](#)  
Group description: Layout/presentation on the WWW.  
 Usenet, 618 subscribers

[netscape.public.mozilla.xpinstall](#) - [join](#)  
Group description: Mozilla XPInstall discussion.  
 Usenet, 22 subscribers

[mozilla.dev.general](#) - [join](#)  
 Usenet, 66 subscribers



User agents may combine the value of the `title` attribute and the other attributes to provide context-sensitive help or inline text detailing the actual values.

For example, the following snippet:

```
<meter min=0 max=60 value=23.2 title=seconds></meter>
```

...might cause the user agent to display a gauge with a tooltip saying "Value: 23.2 out of 60." on one line and "seconds" on a second line.

The `value` IDL attribute, on getting, must return the [actual value](#). On setting, the given value must be converted to the [best representation of the number as a floating-point number](#) and then the `value` content attribute must be set to that string.

The `min` IDL attribute, on getting, must return the [minimum value](#). On setting, the given value must be converted to the [best representation of the number as a floating-point number](#) and then the `min` content attribute must be set to that string.

The `max` IDL attribute, on getting, must return the [maximum value](#). On setting, the given value must be converted to the [best representation of the number as a floating-point number](#) and then the `max` content attribute must be set to that string.

The `low` IDL attribute, on getting, must return the [low boundary](#). On setting, the given value must be converted to the [best representation of the number as a floating-point number](#) and then the `low` content attribute must be set to that string.

The `high` IDL attribute, on getting, must return the [high boundary](#). On setting, the given value must be converted to the [best representation of the number as a floating-point number](#) and then the `high` content attribute must be set to that string.

The `optimum` IDL attribute, on getting, must return the [optimum value](#). On setting, the given value must be converted to the [best representation of the number as a floating-point number](#) and then the `optimum` content attribute must be set to that string.

The `labels` IDL attribute provides a list of the element's `labels`.

The following example shows how a gauge could fall back to localized or pretty-printed text.

```
<p>Disk usage: <meter min=0 value=170261928 max=233257824>170 261 928  
bytes used  
out of 233257824 bytes available</meter></p>
```

#### 4.10.15 The `fieldset` element



## Categories:

[Flow content](#).

[Listed](#) and [autocapitalize-inheriting form-associated element](#).

[Palpable content](#).

## Contexts in which this element can be used:

Where [flow content](#) is expected.

## Content model:

Optionally a [legend](#) element, followed by [flow content](#).

## Tag omission in text/html:

Neither tag is omissible.

## Content attributes:

[Global attributes](#)

[disabled](#) — Whether the descendant form controls, except any inside [legend](#), are disabled

[form](#) — Associates the element with a [form](#) element

[name](#) — Name of the element to use in the [form.elements](#) API.

## Accessibility considerations:

[For authors](#).

[For implementers](#).

## DOM interface:

[Exposed=Window]

```
interface HTMLFieldSetElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute boolean disabled;
```

```
    readonly attribute HTMLFormElement? form;
```

```
    [CEReactions] attribute DOMString name;
```

```
    readonly attribute DOMString type;
```

```
    [SameObject] readonly attribute HTMLCollection elements;
```

```
readonly attribute boolean willValidate;
```

```
[SameObject] readonly attribute ValidityState validity;
```

```
readonly attribute DOMString validationMessage;
```

```
boolean checkValidity();
```

```
boolean reportValidity();
```

```
undefined setCustomValidity(DOMString error);
```

```
};
```

The [fieldset](#) element [represents](#) a set of form controls (or other content) grouped together, optionally with a caption. The caption is given by the first [legend](#) element that is a child of the [fieldset](#) element, if any. The remainder of the descendants form the group.



The **disabled** attribute, when specified, causes all the form control descendants of the [fieldset](#) element, excluding those that are descendants of the [fieldset](#) element's first [legend](#) element child, if any, to be [disabled](#).

A [fieldset](#) element is a **disabled fieldset** if it matches any of the following conditions:

- Its [disabled](#) attribute is specified
- It is a descendant of another [fieldset](#) element whose [disabled](#) attribute is specified, and is *not* a descendant of that [fieldset](#) element's first [legend](#) element child, if any.

The [form](#) attribute is used to explicitly associate the [fieldset](#) element with its [form owner](#). The [name](#) attribute represents the element's name.

#### **[fieldset.type](#)**

Returns the string "fieldset".

#### **[fieldset.elements](#)**

Returns an [HTMLCollection](#) of the form controls in the element.

The **disabled** IDL attribute must [reflect](#) the content attribute of the same name.

The **type** IDL attribute must return the string "fieldset".

The **elements** IDL attribute must return an [HTMLCollection](#) rooted at the [fieldset](#) element, whose filter matches [listed elements](#).

The [willValidate](#), [validity](#), and [validationMessage](#) attributes, and the [checkValidity\(\)](#), [reportValidity\(\)](#), and [setCustomValidity\(\)](#) methods, are part of the [constraint validation API](#). The [form](#) and [name](#) IDL attributes are part of the element's forms API.

This example shows a [fieldset](#) element being used to group a set of related controls:

```
<fieldset>
  <legend>Display</legend>
  <p><label><input type=radio name=c value=0 checked> Black on
White</label>
  <p><label><input type=radio name=c value=1> White on Black</label>
  <p><label><input type=checkbox name=g> Use grayscale</label>
  <p><label>Enhance contrast <input type=range name=e list=contrast
min=0 max=100 value=0 step=1></label>
  <datalist id=contrast>
    <option label=Normal value=0>
    <option label=Maximum value=100>
  </datalist>
</fieldset>
```

The following snippet shows a fieldset with a checkbox in the legend that controls whether or not the fieldset is enabled. The contents of the fieldset consist of two required text controls and an optional year/month control.

```
<fieldset name="clubfields" disabled>
  <legend> <label>
    <input type=checkbox name=club onchange="form.clubfields.disabled
= !checked">
    Use Club Card
  </label> </legend>
  <p><label>Name on card: <input name=clubname required></label></p>
  <p><label>Card number: <input name=clubnum required pattern="[-0-
9]+"></label></p>
  <p><label>Expiry date: <input name=clubexp type=month></label></p>
</fieldset>
```

You can also nest [fieldset](#) elements. Here is an example expanding on the previous one that does so:

```
<fieldset name="clubfields" disabled>
```

```

<legend> <label>
  <input type=checkbox name=club onchange="form.clubfields.disabled
= !checked">
  Use Club Card
</label> </legend>
<p><label>Name on card: <input name=clubname required></label></p>
<fieldset name="numfields">
  <legend> <label>
    <input type=radio checked name=clubtype
onchange="form.numfields.disabled = !checked">
    My card has numbers on it
  </label> </legend>
  <p><label>Card number: <input name=clubnum required pattern="[-0-
9]+"></label></p>
</fieldset>
<fieldset name="letfields" disabled>
  <legend> <label>
    <input type=radio name=clubtype
onchange="form.letfields.disabled = !checked">
    My card has letters on it
  </label> </legend>
  <p><label>Card code: <input name=clublet required pattern="[A-Za-
z]+"></label></p>
</fieldset>
</fieldset>

```

In this example, if the outer "Use Club Card" checkbox is not checked, everything inside the outer fieldset, including the two radio buttons in the legends of the two nested fieldsets, will be disabled. However, if the checkbox is checked, then the radio buttons will both be enabled and will let you select which of the two inner fieldsets is to be enabled.

This example shows a grouping of controls where the legend element both labels the grouping, and the nested heading element surfaces the grouping in the document outline:

```

<fieldset>
  <legend> <h2>
    How can we best reach you?
  </h2> </legend>
  <p> <label>
    <input type=radio checked name=contact_pref>
    Phone
  </label> </p>
  <p> <label>

```

```

☐

```

#### 4.10.16 The **legend** element



##### Categories:

None.

##### Contexts in which this element can be used:

As the first child of a fieldset element.

##### Content model:

Phrasing content, optionally intermixed with heading content.

##### Tag omission in text/html:

Neither tag is omissible.

##### Content attributes:

Global attributes

##### Accessibility considerations:

For authors.

For implementers.

##### DOM interface:

[Exposed=Window]

```
interface HTMLLegendElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    readonly attribute HTMLFormElement? form;
```

```
// also has obsolete members
```

```
};
```

The [legend](#) element [represents](#) a caption for the rest of the contents of the [legend](#) element's parent [fieldset](#) element, if any.

#### **[legend](#).[form](#)**

Returns the element's [form](#) element, if any, or null otherwise.

The [form](#) IDL attribute's behavior depends on whether the [legend](#) element is in a [fieldset](#) element or not. If the [legend](#) has a [fieldset](#) element as its parent, then the [form](#) IDL attribute must return the same value as the [form](#) IDL attribute on that [fieldset](#) element. Otherwise, it must return null.

## 4.10.17 Form control infrastructure

### 4.10.17.1 A form control's value

Most form controls have a **value** and a **checkedness**. (The latter is only used by [input](#) elements.) These are used to describe how the user interacts with the control.

A control's [value](#) is its internal state. As such, it might not match the user's current input.

For instance, if a user enters the word "three" into [a numeric field](#) that expects digits, the user's input would be the string "three" but the control's [value](#) would remain unchanged. Or, if a user enters the email address " [awesome@example.com](#)" (with leading whitespace) into [an email field](#), the user's input would be the string " [awesome@example.com](#)" but the browser's UI for email fields might translate that into a [value](#) of "[awesome@example.com](#)" (without the leading whitespace).

[input](#) and [textarea](#) elements have a **dirty value flag**. This is used to track the interaction between the [value](#) and default value. If it is false, [value](#) mirrors the default value. If it is true, the default value is ignored.

To define the behavior of constraint validation in the face of the [input](#) element's [multiple](#) attribute, [input](#) elements can also have separately defined **values**.

To define the behavior of the [maxlength](#) and [minlength](#) attributes, as well as other APIs specific to the [textarea](#) element, all form control with a [value](#) also have

an algorithm for obtaining an **API value**. By default this algorithm is to simply return the control's [value](#).

The [select](#) element does not have a [value](#); the [selectedness](#) of its [option](#) elements is what is used instead.

#### 4.10.17.2 Mutability

A form control can be designated as **mutable**.

*This determines (by means of definitions and requirements in this specification that rely on whether an element is so designated) whether or not the user can modify the [value](#) or [checkedness](#) of a form control, or whether or not a control can be automatically prefilled.*

#### 4.10.17.3 Association of controls and forms

A [form-associated element](#) can have a relationship with a [form](#) element, which is called the element's **form owner**. If a [form-associated element](#) is not associated with a [form](#) element, its [form owner](#) is said to be null.

A [form-associated element](#) has an associated **parser inserted flag**.



A [form-associated element](#) is, by default, associated with its nearest ancestor [form](#) element (as described below), but, if it is [listed](#), may have a **form** attribute specified to override this.

*This feature allows authors to work around the lack of support for nested [form](#) elements.*

If a [listed form-associated element](#) has a [form](#) attribute specified, then that attribute's value must be the [ID](#) of a [form](#) element in the element's [tree](#).

*The rules in this section are complicated by the fact that although conforming documents or [trees](#) will never contain nested [form](#) elements, it is quite possible (e.g., using a script that performs DOM manipulation) to generate [trees](#) that have such nested elements. They are also complicated by rules in the HTML parser that, for historical reasons, can result in a [form-associated element](#) being associated with a [form](#) element that is not its ancestor.*



When a [form-associated element](#) is created, its [form owner](#) must be initialized to null (no owner).

When a [form-associated element](#) is to be **associated** with a form, its [form owner](#) must be set to that form.

When a [listed form-associated element](#)'s [form](#) attribute is set, changed, or removed, then the user agent must [reset the form owner](#) of that element.

When a [listed form-associated element](#) has a [form](#) attribute and the [ID](#) of any of the elements in the [tree](#) changes, then the user agent must [reset the form owner](#) of that [form-associated element](#).

When a [listed form-associated element](#) has a [form](#) attribute and an element with an [ID](#) is [inserted into](#) or [removed from](#) the [Document](#), then the user agent must [reset the form owner](#) of that [form-associated element](#).

*The form owner is also reset by the HTML Standard's [insertion steps](#) and [removing steps](#).*

When the user agent is to **reset the form owner** of a [form-associated element](#), it must run the following steps:

1. Unset *element*'s [parser inserted flag](#).
2. If all of the following conditions are true
  - *element*'s [form owner](#) is not null
  - *element* is not [listed](#) or its [form](#) content attribute is not present
  - *element*'s [form owner](#) is its nearest [form](#) element ancestor after the change to the ancestor chainthen do nothing, and return.
3. Set *element*'s [form owner](#) to null.
4. If *element* is [listed](#), has a [form](#) content attribute, and is [connected](#), then:
  - If the first element in *element*'s [tree](#), in [tree order](#), to have an [ID](#) that is [identical to](#) *element*'s [form](#) content attribute's value, is a [form](#) element, then [associate](#) the *element* with that [form](#) element.
5. Otherwise, if *element* has an ancestor [form](#) element, then [associate](#) *element* with the nearest such ancestor [form](#) element.

In the following non-conforming snippet

```
...  
<form id="a">
```

```

<div id="b"></div>
</form>
<script>
  document.getElementById('b').innerHTML =
    '<table><tr><td></form><form id="c"><input id="d"></table>' +
    '<input id="e">';
</script>
...

```

the [form owner](#) of "d" would be the inner nested form "c", while the [form owner](#) of "e" would be the outer form "a".

This happens as follows: First, the "e" node gets associated with "c" in the [HTML parser](#). Then, the [innerHTML](#) algorithm moves the nodes from the temporary document to the "b" element. At this point, the nodes see their ancestor chain change, and thus all the "magic" associations done by the parser are reset to normal ancestor associations.

This example is a non-conforming document, though, as it is a violation of the content models to nest [form](#) elements, and there is a [parse error](#) for the `</form>` tag.

**`element.form`**



Returns the element's [form owner](#).

Returns null if there isn't one.

[Listed form-associated elements](#) except for [form-associated custom elements](#) have a [form](#) IDL attribute, which, on getting, must return the element's [form owner](#), or null if there isn't one.



[Form-associated custom elements](#) don't have [form](#) IDL attribute. Instead, their [ElementInternals](#) object has a [form](#) IDL attribute. On getting, it must throw a ["NotSupportedError" DOMException](#) if the [target element](#) is not a [form-associated custom element](#). Otherwise, it must return the element's [form owner](#), or null if there isn't one.

## 4.10.18 Attributes common to form controls

### 4.10.18.1 Naming form controls: the [name](#) attribute



The **name** content attribute gives the name of the form control, as used in [form submission](#) and in the [form](#) element's [elements](#) object. If the attribute is specified, its value must not be the empty string or `isindex`.

*A number of user agents historically implemented special support for first-in-form text controls with the name `isindex`, and this specification previously defined related user agent requirements for it. However, some user agents subsequently dropped that special support, and the related requirements were removed from this specification. So, to avoid problematic reinterpretations in legacy user agents, the name `isindex` is no longer allowed.*

Other than `isindex`, any non-empty value for **name** is allowed. An [ASCII case-insensitive](#) match for the name `_charset_` is special: if used as the name of a [Hidden](#) control with no **value** attribute, then during submission the **value** attribute is automatically given a value consisting of the submission character encoding.

The **name** IDL attribute must [reflect](#) the **name** content attribute.

*DOM clobbering is a common cause of security issues. Avoid using the names of built-in form properties with the **name** content attribute.*

*In this example, the [input](#) element overrides the built-in [method](#) property:*

```
let form = document.createElement("form");
let input = document.createElement("input");
form.appendChild(input);

form.method; // => "get"
input.name = "method"; // DOM clobbering occurs here
form.method === input; // => true
```

*Since the input name takes precedence over built-in form properties, the JavaScript reference `form.method` will point to the [input](#) element named "method" instead of the built-in [method](#) property.*

#### 4.10.18.2 Submitting element directionality: the **dirname** attribute

MDN

The **dirname** attribute on a form control element enables the submission of [the directionality](#) of the element, and gives the name of the control that contains this value during [form submission](#). If such an attribute is specified, its value must not be the empty string.

In this example, a form contains a text control and a submission button:

```
<form action="addcomment.cgi" method=post>
  <p><label>Comment: <input type=text name="comment"
dirname="comment.dir" required></label></p>
  <p><button name="mode" type=submit value="add">Post
Comment</button></p>
</form>
```

When the user submits the form, the user agent includes three fields, one called "comment", one called "comment.dir", and one called "mode"; so if the user types "Hello", the submission body might be something like:

```
comment=Hello&comment.dir=ltr&mode=add
```

If the user manually switches to a right-to-left writing direction and enters "مرحبا", the submission body might be something like:

```
comment=%D9%85%D8%B1%D8%AD%D8%A8%D8%A7&comment.dir=rtl&mode=add
```

#### 4.10.18.3 Limiting user input length: the [maxlength](#) attribute

A **form control** [maxlength](#) attribute, controlled by the [dirty value flag](#), declares a limit on the number of characters a user can input. The number of characters is measured using [length](#) and, in the case of [textarea](#) elements, with all newlines normalized to a single character (as opposed to CRLF pairs).

If an element has its [form control](#) [maxlength](#) attribute specified, the attribute's value must be a [valid non-negative integer](#). If the attribute is specified and applying the [rules for parsing non-negative integers](#) to its value results in a number, then that number is the element's **maximum allowed value length**. If the attribute is omitted or parsing its value results in an error, then there is no [maximum allowed value length](#).

**Constraint validation:** If an element has a [maximum allowed value length](#), its [dirty value flag](#) is true, its [value](#) was last changed by a user edit (as opposed to a change made by a script), and the [length](#) of the element's [API value](#) is greater than the element's [maximum allowed value length](#), then the element is [suffering from being too long](#).

User agents may prevent the user from causing the element's [API value](#) to be set to a value whose [length](#) is greater than the element's [maximum allowed value length](#).

*In the case of [textarea](#) elements, the [API value](#) and [value](#) differ. In particular, [newline normalization](#) is applied before the [maximum allowed value length](#) is checked (whereas the [textarea wrapping transformation](#) is not applied).*

#### 4.10.18.4 Setting minimum input length requirements: the `minlength` attribute

A **form control** `minlength` attribute, controlled by the [dirty value flag](#), declares a lower bound on the number of characters a user can input. The "number of characters" is measured using [length](#) and, in the case of `textarea` elements, with all newlines normalized to a single character (as opposed to CRLF pairs).

*The `minlength` attribute does not imply the `required` attribute. If the form control has no `required` attribute, then the value can still be omitted; the `minlength` attribute only kicks in once the user has entered a value at all. If the empty string is not allowed, then the `required` attribute also needs to be set.*

If an element has its [form control](#) `minlength` attribute specified, the attribute's value must be a [valid non-negative integer](#). If the attribute is specified and applying the [rules for parsing non-negative integers](#) to its value results in a number, then that number is the element's **minimum allowed value length**. If the attribute is omitted or parsing its value results in an error, then there is no [minimum allowed value length](#).

If an element has both a [maximum allowed value length](#) and a [minimum allowed value length](#), the [minimum allowed value length](#) must be smaller than or equal to the [maximum allowed value length](#).

**Constraint validation:** If an element has a [minimum allowed value length](#), its [dirty value flag](#) is true, its [value](#) was last changed by a user edit (as opposed to a change made by a script), its [value](#) is not the empty string, and the [length](#) of the element's [API value](#) is less than the element's [minimum allowed value length](#), then the element is [suffering from being too short](#).

In this example, there are four text controls. The first is required, and has to be at least 5 characters long. The other three are optional, but if the user fills one in, the user has to enter at least 10 characters.

```
<form action="/events/menu.cgi" method="post">
  <p><label>Name of Event: <input required minlength=5 maxlength=50
name=event></label></p>
  <p><label>Describe what you would like for breakfast, if anything:
    <textarea name="breakfast"
minlength="10"></textarea></label></p>
  <p><label>Describe what you would like for lunch, if anything:
    <textarea name="lunch" minlength="10"></textarea></label></p>
  <p><label>Describe what you would like for dinner, if anything:
    <textarea name="dinner" minlength="10"></textarea></label></p>
  <p><input type=submit value="Submit Request"></p>
</form>
```

#### 4.10.18.5 Enabling and disabling form controls: the [disabled](#) attribute



The [disabled](#) content attribute is a [boolean attribute](#).

*The [disabled](#) attribute for [option](#) elements and the [disabled](#) attribute for [optgroup](#) elements are defined separately.*

A form control is **disabled** if any of the following conditions are met:

1. The element is a [button](#), [input](#), [select](#), [textarea](#), or [form-associated custom element](#), and the [disabled](#) attribute is specified on this element (regardless of its value).
2. The element is a descendant of a [fieldset](#) element whose [disabled](#) attribute is specified, and is *not* a descendant of that [fieldset](#) element's first [legend](#) element child, if any.

A form control that is [disabled](#) must prevent any [click](#) events that are [queued](#) on the [user interaction task source](#) from being dispatched on the element.

**Constraint validation:** If an element is [disabled](#), it is [barred from constraint validation](#).



The [disabled](#) IDL attribute must [reflect](#) the [disabled](#) content attribute.

#### 4.10.18.6 Form submission attributes



**Attributes for form submission** can be specified both on [form](#) elements and on [submit buttons](#) (elements that represent buttons that submit forms, e.g. an [input](#) element whose [type](#) attribute is in the [Submit Button](#) state).

The [attributes for form submission](#) that may be specified on [form](#) elements are [action](#), [enctype](#), [method](#), [novalidate](#), and [target](#).

The corresponding [attributes for form submission](#) that may be specified on [submit buttons](#) are [formaction](#), [formenctype](#), [formmethod](#), [formnovalidate](#), and [formtarget](#). When omitted, they default to the values given on the corresponding attributes on the [form](#) element.



The **action** and **formaction** content attributes, if specified, must have a value that is a [valid non-empty URL potentially surrounded by spaces](#).

The **action** of an element is the value of the element's **formaction** attribute, if the element is a [submit button](#) and has such an attribute, or the value of its [form owner](#)'s **action** attribute, if *it* has one, or else the empty string.

---



The **method** and **formmethod** content attributes are [enumerated attributes](#) with the following keywords and states:

- The keyword **get**, mapping to the state **GET**, indicating the HTTP GET method.
- The keyword **post**, mapping to the state **POST**, indicating the HTTP POST method.
- The keyword **dialog**, mapping to the state **dialog**, indicating that submitting the [form](#) is intended to close the [dialog](#) box in which the form finds itself, if any, and otherwise not submit.

The **method** attribute's [invalid value default](#) and [missing value default](#) are both the [GET](#) state.

The **formmethod** attribute's [invalid value default](#) is the [GET](#) state. It has no [missing value default](#).

The **method** of an element is one of those states. If the element is a [submit button](#) and has a **formmethod** attribute, then the element's **method** is that attribute's state; otherwise, it is the [form owner](#)'s **method** attribute's state.

Here the **method** attribute is used to explicitly specify the default value, ["get"](#), so that the search query is submitted in the URL:

```
<form method="get" action="/search.cgi">
  <p><label>Search terms: <input type=search name=q></label></p>
  <p><input type=submit></p>
</form>
```

On the other hand, here the `method` attribute is used to specify the value `"post"`, so that the user's message is submitted in the HTTP request's body:

```
<form method="post" action="/post-message.cgi">
  <p><label>Message: <input type=text name=m></label></p>
  <p><input type=submit value="Submit message"></p>
</form>
```

In this example, a `form` is used with a `dialog`. The `method` attribute's `"dialog"` keyword is used to have the dialog automatically close when the form is submitted.

```
<dialog id="ship">
  <form method=dialog>
    <p>A ship has arrived in the harbour.</p>
    <button type=submit value="board">Board the ship</button>
    <button type=submit value="call">Call to the captain</button>
  </form>
</dialog>
<script>
  var ship = document.getElementById('ship');
  ship.showModal();
  ship.onclose = function (event) {
    if (ship.returnValue == 'board') {
      // ...
    } else {
      // ...
    }
  };
</script>
```



The **enctype** and **formenctype** content attributes are [enumerated attributes](#) with the following keywords and states:

- The "**application/x-www-form-urlencoded**" keyword and corresponding state.
- The "**multipart/form-data**" keyword and corresponding state.
- The "**text/plain**" keyword and corresponding state.

The **enctype** attribute's [invalid value default](#) and [missing value default](#) are both the [application/x-www-form-urlencoded](#) state.

The **formenctype** attribute's [invalid value default](#) is the [application/x-www-form-urlencoded](#) state. It has no [missing value default](#).

The **enctype** of an element is one of those three states. If the element is a [submit button](#) and has a **formenctype** attribute, then the element's **enctype** is that attribute's state; otherwise, it is the [form owner](#)'s **enctype** attribute's state.



The **target** and **formtarget** content attributes, if specified, must have values that are [valid navigable target names or keywords](#).



The **novalidate** and **formnovalidate** content attributes are [boolean attributes](#). If present, they indicate that the form is not to be validated during submission.

The **no-validate state** of an element is true if the element is a [submit button](#) and the element's **formnovalidate** attribute is present, or if the element's [form owner](#)'s **novalidate** attribute is present, and false otherwise.

This attribute is useful to include "save" buttons on forms that have validation constraints, to allow users to save their progress even though they haven't fully

entered the data in the form. The following example shows a simple form that has two required fields. There are three buttons: one to submit the form, which requires both fields to be filled in; one to save the form so that the user can come back and fill it in later; and one to cancel the form altogether.

```
<form action="editor.cgi" method="post">
  <p><label>Name: <input required name=fn></label></p>
  <p><label>Essay: <textarea required
name=essay></textarea></label></p>
  <p><input type=submit name=submit value="Submit essay"></p>
  <p><input type=submit formnovalidate name=save value="Save
essay"></p>
  <p><input type=submit formnovalidate name=cancel
value="Cancel"></p>
</form>
```



The **action** IDL attribute must [reflect](#) the content attribute of the same name, except that on getting, when the content attribute is missing or its value is the empty string, the element's [node document](#)'s [URL](#) must be returned instead. The **target** IDL attribute must [reflect](#) the content attribute of the same name.

The **method** and **enctype** IDL attributes must [reflect](#) the respective content attributes of the same name, [limited to only known values](#). The **encoding** IDL attribute must [reflect](#) the **enctype** content attribute, [limited to only known values](#).

The **noValidate** IDL attribute must [reflect](#) the **novalidate** content attribute.

The **formAction** IDL attribute must [reflect](#) the **formaction** content attribute, except that on getting, when the content attribute is missing or its value is the empty string, the element's [node document](#)'s [URL](#) must be returned instead. The **formEnctype** IDL attribute must [reflect](#) the **formenctype** content attribute, [limited to only known values](#). The **formMethod** IDL attribute must [reflect](#) the **formmethod** content attribute, [limited to only known values](#). The **formNoValidate** IDL attribute must [reflect](#) the **formnovalidate** content attribute. The **formTarget** IDL attribute must [reflect](#) the **formtarget** content attribute.

## 4.10.18.7 Autofill

### 4.10.18.7.1 Autofilling form controls: the **autocomplete** attribute

User agents sometimes have features for helping users fill forms in, for example prefilling the user's address based on earlier user input. The **autocomplete** content attribute can be used to hint to the user agent how to, or indeed whether to, provide such a feature.

There are two ways this attribute is used. When wearing the **autofill expectation mantle**, the **autocomplete** attribute describes what input is expected from users. When wearing the **autofill anchor mantle**, the **autocomplete** attribute describes the meaning of the given value.

On an **input** element whose **type** attribute is in the **Hidden** state, the **autocomplete** attribute wears the **autofill anchor mantle**. In all other cases, it wears the **autofill expectation mantle**.

When wearing the **autofill expectation mantle**, the **autocomplete** attribute, if specified, must have a value that is an ordered **set of space-separated tokens** consisting of either a single token that is an **ASCII case-insensitive** match for the string "**off**", or a single token that is an **ASCII case-insensitive** match for the string "**on**", or **autofill detail tokens**.

When wearing the **autofill anchor mantle**, the **autocomplete** attribute, if specified, must have a value that is an ordered **set of space-separated tokens** consisting of just **autofill detail tokens** (i.e. the "**on**" and "**off**" keywords are not allowed).

**Autofill detail tokens** are the following, in the order given below:

1. Optionally, a token whose first eight characters are an **ASCII case-insensitive** match for the string "**section-**", meaning that the field belongs to the named group.

For example, if there are two shipping addresses in the form, then they could be marked up as:

```
<fieldset>
  <legend>Ship the blue gift to...</legend>
  <p> <label> Address:      <textarea name=ba
autocomplete="section-blue shipping street-
address"></textarea> </label>
  <p> <label> City:        <input name=bc
autocomplete="section-blue shipping address-level2"> </label>
  <p> <label> Postal Code: <input name=bp
autocomplete="section-blue shipping postal-code"> </label>
</fieldset>
<fieldset>
  <legend>Ship the red gift to...</legend>
```

```

<p> <label> Address:      <textarea name=ra
autocomplete="section-red shipping street-
address"></textarea> </label>

<p> <label> City:         <input name=rc
autocomplete="section-red shipping address-level2"> </label>

<p> <label> Postal Code: <input name=rp
autocomplete="section-red shipping postal-code"> </label>
</fieldset>

```

2. Optionally, a token that is an [ASCII case-insensitive](#) match for one of the following strings:
  - "shipping", meaning the field is part of the shipping address or contact information
  - "billing", meaning the field is part of the billing address or contact information
3. Either of the following two options:
  - A token that is an [ASCII case-insensitive](#) match for one of the following [autofill field](#) names, excluding those that are [inappropriate for the control](#):

- "name"
- "honorific-prefix"
- "given-name"
- "additional-name"
- "family-name"
- "honorific-suffix"
- "nickname"
- "username"
- "new-password"
- "current-password"
- "one-time-code"
- "organization-title"
- "organization"
- "street-address"
- "address-line1"
- "address-line2"
- "address-line3"
- "address-level4"
- "address-level3"
- "address-level2"
- "address-level1"
- "country"
- "country-name"
- "postal-code"
- "cc-name"
- "cc-given-name"

- "cc-additional-name"
- "cc-family-name"
- "cc-number"
- "cc-exp"
- "cc-exp-month"
- "cc-exp-year"
- "cc-csc"
- "cc-type"
- "transaction-currency"
- "transaction-amount"
- "language"
- "bday"
- "bday-day"
- "bday-month"
- "bday-year"
- "sex"
- "url"
- "photo"

(See the table below for descriptions of these values.)

- The following, in the given order:
  - Optionally, a token that is an [ASCII case-insensitive](#) match for one of the following strings:
    - **"home"**, meaning the field is for contacting someone at their residence
    - **"work"**, meaning the field is for contacting someone at their workplace
    - **"mobile"**, meaning the field is for contacting someone regardless of location
    - **"fax"**, meaning the field describes a fax machine's contact details
    - **"pager"**, meaning the field describes a pager's or beeper's contact details
  - A token that is an [ASCII case-insensitive](#) match for one of the following [autofill field](#) names, excluding those that are [inappropriate for the control](#):
    - "tel"
    - "tel-country-code"
    - "tel-national"
    - "tel-area-code"
    - "tel-local"
    - "tel-local-prefix"
    - "tel-local-suffix"
    - "tel-extension"

- "email"
- "impp"

(See the table below for descriptions of these values.)

4. Optionally, a token that is an [ASCII case-insensitive](#) match for the string "webauthn", meaning the user agent should show [public key credentials](#) available via [conditional](#) mediation when the user interacts with the form control. webauthn is only valid for [input](#) and [textarea](#) elements.

As noted earlier, the meaning of the attribute and its keywords depends on the mantle that the attribute is wearing.

### When wearing the [autofill expectation mantle](#)...

The "off" keyword indicates either that the control's input data is particularly sensitive (for example the activation code for a nuclear weapon); or that it is a value that will never be reused (for example a one-time-key for a bank login) and the user will therefore have to explicitly enter the data each time, instead of being able to rely on the UA to prefill the value for them; or that the document provides its own autocomplete mechanism and does not want the user agent to provide autocompletion values.

The "on" keyword indicates that the user agent is allowed to provide the user with autocompletion values, but does not provide any further information about what kind of data the user might be expected to enter. User agents would have to use heuristics to decide what autocompletion values to suggest.

The [autofill field](#) listed above indicate that the user agent is allowed to provide the user with autocompletion values, and specifies what kind of value is expected. The meaning of each such keyword is described in the table below.

If the [autocomplete](#) attribute is omitted, the default value corresponding to the state of the element's [form owner](#)'s [autocomplete](#) attribute is used instead (either "on" or "off"). If there is no [form owner](#), then the value "on" is used.

### When wearing the [autofill anchor mantle](#)...

The [autofill field](#) listed above indicate that the value of the particular kind of value specified is that value provided for this element. The meaning of each such keyword is described in the table below.

In this example the page has explicitly specified the currency and amount of the transaction. The form requests a credit card and other billing details. The user agent could use this information to suggest a credit card that it knows has sufficient balance and that supports the relevant currency.

```
<form method=post action="step2.cgi">
  <input type=hidden autocomplete=transaction-currency
value="CHF">
```

```

<input type=hidden autocomplete=transaction-amount
value="15.00">

<p><label>Credit card number: <input type=text
inputmode=numeric autocomplete=cc-number></label>

<p><label>Expiry Date: <input type=month autocomplete=cc-
exp></label>

<p><input type=submit value="Continue...">
</form>

```

The **autofill field** keywords relate to each other as described in the table below. Each field name listed on a row of this table corresponds to the meaning given in the cell for that row in the column labeled "Meaning". Some fields correspond to subparts of other fields; for example, a credit card expiry date can be expressed as one field giving both the month and year of expiry ("[cc-exp](#)"), or as two fields, one giving the month ("[cc-exp-month](#)") and one the year ("[cc-exp-year](#)"). In such cases, the names of the broader fields cover multiple rows, in which the narrower fields are defined.

*Generally, authors are encouraged to use the broader fields rather than the narrower fields, as the narrower fields tend to expose Western biases. For example, while it is common in some Western cultures to have a given name and a family name, in that order (and thus often referred to as a first name and a surname), many cultures put the family name first and the given name second, and many others simply have one name (a mononym). Having a single field is therefore more flexible.*

Some fields are only appropriate for certain form controls. An [autofill field](#) name is **inappropriate for a control** if the control does not belong to the group listed for that [autofill field](#) in the fifth column of the first row describing that [autofill field](#) in the table below. What controls fall into each group is described below the table.

Field name	Meaning	Canonical Format	Canonical Format Example	Control group
" <a href="#">name</a> "	Full name	Free-form text, no newlines	Sir Timothy John Berners-Lee, OM, KBE, FRS, FREng, FRSA	<a href="#">Text</a>
" <a href="#">honorific-prefix</a> "	Prefix or title (e.g. "Mr.", "Ms.", "Dr.", "Mlle")	Free-form text, no newlines	Sir	<a href="#">Text</a>
" <a href="#">given-name</a> "	Given name (in some Western cultures, also known as the <i>first name</i> )	Free-form text, no newlines	Timothy	<a href="#">Text</a>
" <a href="#">additional-name</a> "	Additional names (in some Western cultures, also known as <i>middle names</i> ,	Free-form text, no newlines	John	<a href="#">Text</a>

Field name	Meaning	Canonical Format	Canonical Format Example	Control group
	forenames other than the first name)			
"family-name"	Family name (in some Western cultures, also known as the <i>last name</i> or <i>surname</i> )	Free-form text, no newlines	Berners-Lee	<a href="#">Text</a>
"honorific-suffix"	Suffix (e.g. "Jr.", "B.Sc.", "MBASW", "II")	Free-form text, no newlines	OM, KBE, FRS, FEng, FRSA	<a href="#">Text</a>
"nickname"	Nickname, screen name, handle: a typically short name used instead of the full name	Free-form text, no newlines	Tim	<a href="#">Text</a>
"organization-title"	Job title (e.g. "Software Engineer", "Senior Vice President", "Deputy Managing Director")	Free-form text, no newlines	Professor	<a href="#">Text</a>
"username"	A username	Free-form text, no newlines	timbl	<a href="#">Username</a>
"new-password"	A new password (e.g. when creating an account or changing a password)	Free-form text, no newlines	GUMFXbadyrS3	<a href="#">Password</a>
"current-password"	The current password for the account identified by the <a href="#">username</a> field (e.g. when logging in)	Free-form text, no newlines	qwerty	<a href="#">Password</a>
"one-time-code"	One-time code used for verifying user identity	Free-form text, no newlines	123456	<a href="#">Password</a>
"organization"	Company name corresponding to the person, address, or contact information in the other fields associated with this field	Free-form text, no newlines	World Wide Web Consortium	<a href="#">Text</a>



Field name	Meaning	Canonical Format	Canonical Format Example	Control group
"street-address"	Street address (multiple lines, newlines preserved)	Free-form text	32 Vassar Street MIT Room 32-G524	<a href="#">Multiline</a>
"address-line1"	Street address (one line per field)	Free-form text, no newlines	32 Vassar Street	<a href="#">Text</a>
"address-line2"		Free-form text, no newlines	MIT Room 32-G524	<a href="#">Text</a>
"address-line3"		Free-form text, no newlines		<a href="#">Text</a>
"address-level4"	The most fine-grained <a href="#">administrative level</a> , in addresses with four administrative levels	Free-form text, no newlines		<a href="#">Text</a>
"address-level3"	The <a href="#">third administrative level</a> , in addresses with three or more administrative levels	Free-form text, no newlines		<a href="#">Text</a>
"address-level2"	The <a href="#">second administrative level</a> , in addresses with two or more administrative levels; in the countries with two administrative levels, this would typically be the city, town, village, or other locality within which the relevant street address is found	Free-form text, no newlines	Cambridge	<a href="#">Text</a>
"address-level1"	The broadest <a href="#">administrative level</a> in the address, i.e. the province within which the locality is found; for example, in the US, this would be the state; in Switzerland it	Free-form text, no newlines	MA	<a href="#">Text</a>

Field name	Meaning	Canonical Format	Canonical Format Example	Control group
	would be the canton; in the UK, the post town			
"country"	Country code	Valid <a href="#">ISO 3166-1-alpha-2 country code [ISO3166]</a>	US	<a href="#">Text</a>
"country-name"	Country name	Free-form text, no newlines; <a href="#">derived from country in some cases</a>	US	<a href="#">Text</a>
"postal-code"	Postal code, post code, ZIP code, CEDEX code (if CEDEX, append "CEDEX", and the <i>arrondissement</i> , if relevant, to the <a href="#">address-level2</a> field)	Free-form text, no newlines	02139	<a href="#">Text</a>
"cc-name"	Full name as given on the payment instrument	Free-form text, no newlines	Tim Berners-Lee	<a href="#">Text</a>
"cc-given-name"	Given name as given on the payment instrument (in some Western cultures, also known as the <i>first name</i> )	Free-form text, no newlines	Tim	<a href="#">Text</a>
"cc-additional-name"	Additional names given on the payment instrument (in some Western cultures, also known as <i>middle names</i> , forenames other than the first name)	Free-form text, no newlines		<a href="#">Text</a>

Field name	Meaning	Canonical Format	Canonical Format Example	Control group
"cc-family-name"	Family name given on the payment instrument (in some Western cultures, also known as the <i>last name</i> or <i>surname</i> )	Free-form text, no newlines	Berners-Lee	<a href="#">Text</a>
"cc-number"	Code identifying the payment instrument (e.g. the credit card number)	<a href="#">ASCII digits</a>	4114360123456785	<a href="#">Text</a>
"cc-exp"	Expiration date of the payment instrument	<a href="#">Valid month string</a>	2014-12	<a href="#">Month</a>
"cc-exp-month"	Month component of the expiration date of the payment instrument	<a href="#">Valid integer</a> in the range 1..12	12	<a href="#">Numeric</a>
"cc-exp-year"	Year component of the expiration date of the payment instrument	<a href="#">Valid integer</a> greater than zero	2014	<a href="#">Numeric</a>
"cc-csc"	Security code for the payment instrument (also known as the card security code (CSC), card validation code (CVC), card verification value (CVV), signature panel code (SPC), credit card ID (CCID), etc.)	<a href="#">ASCII digits</a>	419	<a href="#">Text</a>
"cc-type"	Type of payment instrument	Free-form text, no newlines	Visa	<a href="#">Text</a>
"transaction-currency"	The currency that the user would prefer the transaction to use	ISO 4217 currency code <a href="#">[ISO4217]</a>	GBP	<a href="#">Text</a>
"transaction-amount"	The amount that the user would like for the transaction (e.g.	<a href="#">Valid floating-</a>	401.00	<a href="#">Numeric</a>

Field name	Meaning	Canonical Format	Canonical Format Example	Control group
	when entering a bid or sale price)	<a href="#">point number</a>		
" <b>language</b> "	Preferred language	Valid BCP 47 language tag <a href="#">[BCP47]</a>	en	<a href="#">Text</a>
" <b>bday</b> "	Birthday	<a href="#">Valid date string</a>	1955-06-08	<a href="#">Date</a>
" <b>bday-day</b> "	Day component of birthday	<a href="#">Valid integer</a> in the range 1..31	8	<a href="#">Numeric</a>
" <b>bday-month</b> "	Month component of birthday	<a href="#">Valid integer</a> in the range 1..12	6	<a href="#">Numeric</a>
" <b>bday-year</b> "	Year component of birthday	<a href="#">Valid integer</a> greater than zero	1955	<a href="#">Numeric</a>
" <b>sex</b> "	Gender identity (e.g. Female, Fa'afafine)	Free-form text, no newlines	Male	<a href="#">Text</a>
" <b>url</b> "	Home page or other web page corresponding to the company, person, address, or contact information in the other fields associated with this field	<a href="#">Valid URL string</a>	<a href="https://www.w3.org/People/Berners-Lee/">https://www.w3.org/People/Berners-Lee/</a>	<a href="#">URL</a>
" <b>photo</b> "	Photographie, icône ou autre image correspondant à l'entreprise, à la personne, à l'adresse ou aux coordonnées dans les autres champs associés à ce champ	<a href="#">Chaîne d'URL valide</a>	<a href="https://www.w3.org/Press/Stock/Berners-Lee/2001-europaeum-eighth.jpg">https://www.w3.org/Press/Stock/Berners-Lee/2001-europaeum-eighth.jpg</a>	<a href="#">URL</a>

Field name	Meaning	Canonical Format	Canonical Format Example	Control group
"tel"	Numéro de téléphone complet, y compris l'indicatif du pays	<a href="#">Chiffres ASCII</a> et caractères U+0020 ESPACE, préfixés par un caractère U+002B PLUS SIGN (+)	+1 617 253 5702	<a href="#">Tél</a>
"tel-country-code"	Composante de l'indicatif de pays du numéro de téléphone	<a href="#">Chiffres ASCII</a> préfixés par un caractère U+002B PLUS SIGN (+)	+1	<a href="#">Texte</a>
"tel-national"	Numéro de téléphone sans le composant de code de comté, avec un préfixe interne au pays appliqué le cas échéant	<a href="#">Chiffres ASCII</a> et caractères U+0020 SPACE	617 253 5702	<a href="#">Texte</a>
"tel-area-code"	Composante de l'indicatif régional du numéro de téléphone, avec un préfixe interne au pays appliqué le cas échéant	<a href="#">Chiffres ASCII</a>	617	<a href="#">Texte</a>
"tel-local"	Numéro de téléphone sans les composants d'indicatif de pays et d'indicatif régional	<a href="#">Chiffres ASCII</a>	2535702	<a href="#">Texte</a>
"tel-local-prefix"	Première partie de la composante du numéro de téléphone qui suit l'indicatif régional, lorsque cette composante est divisée en deux composantes	<a href="#">Chiffres ASCII</a>	253	<a href="#">Texte</a>

Field name	Meaning	Canonical Format	Canonical Format Example	Control group
"tel-local-suffix"	Deuxième partie de la composante du numéro de téléphone qui suit l'indicatif régional, lorsque cette composante est divisée en deux composantes	<a href="#">Chiffres ASCII</a>	5702	<a href="#">Texte</a>
"tel-extension"	Code de poste interne du numéro de téléphone	<a href="#">Chiffres ASCII</a>	1000	<a href="#">Texte</a>
"email"	Adresse e-mail	<a href="#">Adresse e-mail valable</a>	timbl@w3.org	<a href="#">Nom d'utilisateur</a>
"impp"	URL représentant un point de terminaison de protocole de messagerie instantanée (par exemple, "aim:goim?screenname=example" ou "xmpp:fred@example.net")	<a href="#">Chaîne d'URL valide</a>	irc://example.org/timbl,issuer	<a href="#">URL</a>

Les groupes correspondent aux contrôles comme suit :

### Texte

[input](#) éléments avec un [type](#) attribut à l'état [Masqué](#)  
[input](#) éléments avec un [type](#) attribut à l'état [Texte](#)  
[input](#) éléments avec un [type](#) attribut dans l'état [Recherche](#)  
[textarea](#) éléments  
[select](#) éléments

### Multiligne

[input](#) éléments avec un [type](#) attribut à l'état [Masqué](#)  
[textarea](#) éléments  
[select](#) éléments

### Mot de passe

[input](#) éléments avec un [type](#) attribut à l'état [Masqué](#)  
[input](#) éléments avec un [type](#) attribut à l'état [Texte](#)  
[input](#) éléments avec un [type](#) attribut dans l'état [Recherche](#)  
[input](#) éléments avec un [type](#) attribut à l'état [Mot de passe](#)

textareaéléments

selectéléments

## **URL**

inputéléments avec un typeattribut à l' état [Masqué](#)

inputéléments avec un typeattribut à l' état [Texte](#)

inputéléments avec un typeattribut dans l' état [Recherche](#)

inputéléments avec un typeattribut dans l' état [de l'URL](#)

textareaéléments

selectéléments

## **Nom d'utilisateur**

inputéléments avec un typeattribut à l' état [Masqué](#)

inputéléments avec un typeattribut à l' état [Texte](#)

inputéléments avec un typeattribut dans l' état [Recherche](#)

inputles éléments avec un typeattribut dans l' état [Email](#)

textareaéléments

selectéléments

## **Tél**

inputéléments avec un typeattribut à l' état [Masqué](#)

inputéléments avec un typeattribut à l' état [Texte](#)

inputéléments avec un typeattribut dans l' état [Recherche](#)

inputéléments avec un typeattribut dans l' état [Téléphone](#)

textareaéléments

selectéléments

## **Numérique**

inputéléments avec un typeattribut à l' état [Masqué](#)

inputéléments avec un typeattribut à l' état [Texte](#)

inputéléments avec un typeattribut dans l' état [Recherche](#)

input elements with a type attribute in the [Number](#) state

textarea elements

select elements

## **Month**

input elements with a type attribute in the [Hidden](#) state

input elements with a type attribute in the [Text](#) state

input elements with a type attribute in the [Search](#) state

input elements with a type attribute in the [Month](#) state

textarea elements

select elements

## **Date**

input elements with a type attribute in the [Hidden](#) state

input elements with a type attribute in the [Text](#) state

input elements with a type attribute in the [Search](#) state

input elements with a type attribute in the [Date](#) state

[textarea](#) elements

[select](#) elements

**Address levels:** The "[address-level1](#)" – "[address-level4](#)" fields are used to describe the locality of the street address. Different locales have different numbers of levels. For example, the US uses two levels (state and town), the UK uses one or two depending on the address (the post town, and in some cases the locality), and China can use three (province, city, district). The "[address-level1](#)" field represents the widest administrative division. Different locales order the fields in different ways; for example, in the US the town (level 2) precedes the state (level 1); while in Japan the prefecture (level 1) precedes the city (level 2) which precedes the district (level 3). Authors are encouraged to provide forms that are presented in a way that matches the country's conventions (hiding, showing, and rearranging fields accordingly as the user changes the country).

#### 4.10.18.7.2 Processing model

Each [input](#) element to which the [autocomplete](#) attribute [applies](#), each [select](#) element, and each [textarea](#) element, has an **autofill hint set**, an **autofill scope**, an **autofill field name**, a **non-autofill credential type**, and an **IDL-exposed autofill value**.

The [autofill field name](#) specifies the specific kind of data expected in the field, e.g. "[street-address](#)" or "[cc-exp](#)".

The [autofill hint set](#) identifies what address or contact information type the user agent is to look at, e.g. "[shipping fax](#)" or "[billing](#)".

The [non-autofill credential type](#) identifies a type of [credential](#) that may be offered by the user agent when the user interacts with the field alongside other [autofill field](#) values. If this value is "webauthn" instead of null, selecting a credential of that type will resolve a pending [conditional](#) mediation [navigator.credentials.get\(\)](#) request, instead of autofilling the field.

For example, a sign-in page could instruct the user agent to either autofill a saved password, or show a [public key credential](#) that will resolve a pending [navigator.credentials.get\(\)](#) request. A user can select either to sign-in.

```
<input name=password type=password autocomplete="current-password webauthn">
```

The [autofill scope](#) identifies the group of fields whose information concerns the same subject, and consists of the [autofill hint set](#) with, if applicable, the "section-\*" prefix, e.g. "billing", "section-parent shipping", or "section-child shipping home".



These values are defined as the result of running the following algorithm:

1. If the element has no [autocomplete](#) attribute, then jump to the step labeled *default*.
2. Let *tokens* be the result of [splitting the attribute's value on ASCII whitespace](#).
3. If *tokens* is empty, then jump to the step labeled *default*.
4. Let *index* be the index of the last token in *tokens*.
5. Let *field* be the *index*th token in *tokens*.
6. Set the *category*, *maximum tokens* pair to the result of [determining a field's category](#) given *field*.
7. If *category* is null, then jump to the step labeled *default*.
8. If the number of tokens in *tokens* is greater than *maximum tokens*, then jump to the step labeled *default*.
9. If *category* is Off or Automatic but the element's [autocomplete](#) attribute is wearing the [autofill anchor mantle](#), then jump to the step labeled *default*.
10. If *category* is Off, let the element's [autofill field name](#) be the string "off", let its [autofill hint set](#) be empty, and let its [IDL-exposed autofill value](#) be the string "off". Then, return.
11. If *category* is Automatic, let the element's [autofill field name](#) be the string "on", let its [autofill hint set](#) be empty, and let its [IDL-exposed autofill value](#) be the string "on". Then, return.
12. Let *scope tokens* be an empty list.
13. Let *hint tokens* be an empty set.
14. Let *credential type* be null.
15. Let *IDL value* have the same value as *field*.
16. If *category* is Credential and the *index*th token in *tokens* is an [ASCII case-insensitive](#) match for "[webauthn](#)", then run the substeps that follow:
  1. Set *credential type* to "webauthn".
  2. If the *index*th token in *tokens* is the first entry, then skip to the step labeled *done*.
  3. Decrement *index* by one.
  4. Set the *category*, *maximum tokens* pair to the result of [determining a field's category](#) given the *index*th token in *tokens*.

5. If *category* is not Normal and *category* is not Contact, then jump to the step labeled *default*.
  6. If *index* is greater than *maximum tokens* minus one (i.e. if the number of remaining tokens is greater than *maximum tokens*), then jump to the step labeled *default*.
  7. Set *IDL value* to the concatenation of the *indexth* token in *tokens*, a U+0020 SPACE character, and the previous value of *IDL value*.
17. If the *indexth* token in *tokens* is the first entry, then skip to the step labeled *done*.
18. Decrement *index* by one.
19. If *category* is Contact and the *indexth* token in *tokens* is an [ASCII case-insensitive](#) match for one of the strings in the following list, then run the substeps that follow:
1. "home"
  2. "work"
  3. "mobile"
  4. "fax"
  5. "pager"

The substeps are:

6. Let *contact* be the matching string from the list above.
  7. Insert *contact* at the start of *scope tokens*.
  8. Add *contact* to *hint tokens*.
  9. Let *IDL value* be the concatenation of *contact*, a U+0020 SPACE character, and the previous value of *IDL value*.
  10. If the *indexth* entry in *tokens* is the first entry, then skip to the step labeled *done*.
  11. Decrement *index* by one.
20. If the *indexth* token in *tokens* is an [ASCII case-insensitive](#) match for one of the strings in the following list, then run the substeps that follow:
1. "shipping"
  2. "billing"

The substeps are:

3. Let *mode* be the matching string from the list above.
4. Insert *mode* at the start of *scope tokens*.

5. Add *mode* to *hint tokens*.
6. Let *IDL value* be the concatenation of *mode*, a U+0020 SPACE character, and the previous value of *IDL value*.
7. If the *indexth* entry in *tokens* is the first entry, then skip to the step labeled *done*.
8. Decrement *index* by one.
21. If the *indexth* entry in *tokens* is not the first entry, then jump to the step labeled *default*.
22. If the first eight characters of the *indexth* token in *tokens* are not an [ASCII case-insensitive](#) match for the string "[section-](#)", then jump to the step labeled *default*.
23. Let *section* be the *indexth* token in *tokens*, [converted to ASCII lowercase](#).
24. Insert *section* at the start of *scope tokens*.
25. Let *IDL value* be the concatenation of *section*, a U+0020 SPACE character, and the previous value of *IDL value*.
26. *Done*: Let the element's [autofill hint set](#) be *hint tokens*.
27. Let the element's [non-autofill credential type](#) be *credential type*.
28. Let the element's [autofill scope](#) be *scope tokens*.
29. Let the element's [autofill field name](#) be *field*.
30. Let the element's [IDL-exposed autofill value](#) be *IDL value*.
31. Return.
32. *Default*: Let the element's [IDL-exposed autofill value](#) be the empty string, and its [autofill hint set](#) and [autofill scope](#) be empty.
33. If the element's [autocomplete](#) attribute is wearing the [autofill anchor mantle](#), then let the element's [autofill field name](#) be the empty string and return.
34. Let *form* be the element's [form owner](#), if any, or null otherwise.
35. If *form* is not null and *form*'s [autocomplete](#) attribute is in the [off](#) state, then let the element's [autofill field name](#) be "[off](#)".  
  
Otherwise, let the element's [autofill field name](#) be "[on](#)".

To **determine a field's category**, given *field*:

1. If the *field* is not an [ASCII case-insensitive](#) match for one of the tokens given in the first column of the following table, return the pair (null, null).

Token	Maximum number of tokens	Category
" <a href="#">off</a> "	1	Off
" <a href="#">on</a> "	1	Automatic
" <a href="#">name</a> "	3	Normal
" <a href="#">honorific-prefix</a> "	3	Normal
" <a href="#">given-name</a> "	3	Normal
" <a href="#">additional-name</a> "	3	Normal
" <a href="#">family-name</a> "	3	Normal
" <a href="#">honorific-suffix</a> "	3	Normal
" <a href="#">nickname</a> "	3	Normal
" <a href="#">organization-title</a> "	3	Normal
" <a href="#">username</a> "	3	Normal
" <a href="#">new-password</a> "	3	Normal
" <a href="#">current-password</a> "	3	Normal
" <a href="#">one-time-code</a> "	3	Normal
" <a href="#">organization</a> "	3	Normal
" <a href="#">street-address</a> "	3	Normal
" <a href="#">address-line1</a> "	3	Normal
" <a href="#">address-line2</a> "	3	Normal
" <a href="#">address-line3</a> "	3	Normal
" <a href="#">address-level4</a> "	3	Normal
" <a href="#">address-level3</a> "	3	Normal
" <a href="#">address-level2</a> "	3	Normal
" <a href="#">address-level1</a> "	3	Normal
" <a href="#">country</a> "	3	Normal
" <a href="#">country-name</a> "	3	Normal
" <a href="#">postal-code</a> "	3	Normal
" <a href="#">cc-name</a> "	3	Normal
" <a href="#">cc-given-name</a> "	3	Normal
" <a href="#">cc-additional-name</a> "	3	Normal
" <a href="#">cc-family-name</a> "	3	Normal
" <a href="#">cc-number</a> "	3	Normal
" <a href="#">cc-exp</a> "	3	Normal

Token	Maximum number of tokens	Category
" <u>cc-exp-month</u> "	3	Normal
" <u>cc-exp-year</u> "	3	Normal
" <u>cc-csc</u> "	3	Normal
" <u>cc-type</u> "	3	Normal
" <u>transaction-currency</u> "	3	Normal
" <u>transaction-amount</u> "	3	Normal
" <u>language</u> "	3	Normal
" <u>bday</u> "	3	Normal
" <u>bday-day</u> "	3	Normal
" <u>bday-month</u> "	3	Normal
" <u>bday-year</u> "	3	Normal
" <u>sex</u> "	3	Normal
" <u>url</u> "	3	Normal
" <u>photo</u> "	3	Normal
" <u>tel</u> "	4	Contact
" <u>tel-country-code</u> "	4	Contact
" <u>tel-national</u> "	4	Contact
" <u>tel-area-code</u> "	4	Contact
" <u>tel-local</u> "	4	Contact
" <u>tel-local-prefix</u> "	4	Contact
" <u>tel-local-suffix</u> "	4	Contact
" <u>tel-extension</u> "	4	Contact
" <u>email</u> "	4	Contact
" <u>impp</u> "	4	Contact
" <u>webauthn</u> "	5	Credential

- Otherwise, let *maximum tokens* and *category* be the values of the cells in the second and third columns of that row respectively.
- Return the pair (*category*, *maximum tokens*).

---

For the purposes of autofill, a **control's data** depends on the kind of control:

An input element with its type attribute in the Email state and with the multiple attribute specified

The element's values.

Any other input element

A textarea element

The element's value.

A select element with its multiple attribute specified

The option elements in the select element's list of options that have their selectedness set to true.

Any other select element

The option element in the select element's list of options that has its selectedness set to true.

---

How to process the autofill hint set, autofill scope, and autofill field name depends on the mantle that the autocomplete attribute is wearing.

When wearing the autofill expectation mantle...

When an element's autofill field name is "off", the user agent should not remember the control's data, and should not offer past values to the user.

*In addition, when an element's autofill field name is "off", values are reset when reactivating a document.*

Banks frequently do not want UAs to prefill login information:

```
<p><label>Account: <input type="text" name="ac"
autocomplete="off"></label></p>

<p><label>PIN: <input type="password" name="pin"
autocomplete="off"></label></p>
```

When an element's autofill field name is *not* "off", the user agent may store the control's data, and may offer previously stored values to the user.

For example, suppose a user visits a page with this control:

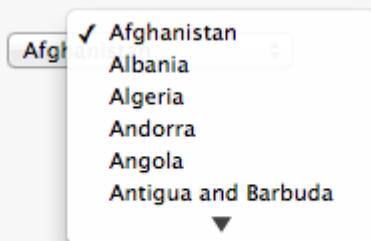
```
<select name="country">
  <option>Afghanistan
  <option>Albania
  <option>Algeria
  <option>Andorra
  <option>Angola
```

```

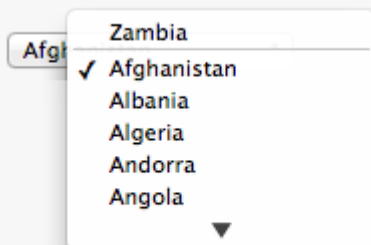
<option>Antigua and Barbuda
<option>Argentina
<option>Armenia
<!-- ... -->
<option>Yemen
<option>Zambia
<option>Zimbabwe
</select>

```

This might render as follows:



Suppose that on the first visit to this page, the user selects "Zambia". On the second visit, the user agent could duplicate the entry for Zambia at the top of the list, so that the interface instead looks like this:



When the [autofill field name](#) is "[on](#)", the user agent should attempt to use heuristics to determine the most appropriate values to offer the user, e.g. based on the element's [name](#) value, the position of the element in its [tree](#), what other fields exist in the form, and so forth.

When the [autofill field name](#) is one of the names of the [autofill fields](#) described above, the user agent should provide suggestions that match the meaning of the field name as given in the table earlier in this section. The [autofill hint set](#) should be used to select amongst multiple possible suggestions.

For example, if a user once entered one address into fields that used the "[shipping](#)" keyword, and another address into fields that used the "[billing](#)" keyword, then in subsequent forms only the first address would be

suggested for form controls whose [autofill hint set](#) contains the keyword ["shipping"](#). Both addresses might be suggested, however, for address-related form controls whose [autofill hint set](#) does not contain either keyword.

### When wearing the [autofill anchor mantle](#)...

When the [autofill field name](#) is not the empty string, then the user agent must act as if the user had specified the [control's data](#) for the given [autofill hint set](#), [autofill scope](#), and [autofill field name](#) combination.

When the user agent **autofills form controls**, elements with the same [form owner](#) and the same [autofill scope](#) must use data relating to the same person, address, payment instrument, and contact details. When a user agent autofills ["country"](#) and ["country-name"](#) fields with the same [form owner](#) and [autofill scope](#), and the user agent has a value for the [country](#) field(s), then the ["country-name"](#) field(s) must be filled using a human-readable name for the same country. When a user agent fills in multiple fields at once, all fields with the same [autofill field name](#), [form owner](#) and [autofill scope](#) must be filled with the same value.

Suppose a user agent knows of two phone numbers, +1 555 123 1234 and +1 555 666 7777. It would not be conforming for the user agent to fill a field with `autocomplete="shipping tel-local-prefix"` with the value "123" and another field in the same form with `autocomplete="shipping tel-local-suffix"` with the value "7777". The only valid prefilled values given the aforementioned information would be "123" and "1234", or "666" and "7777", respectively. Similarly, if a form for some reason contained both a ["cc-exp"](#) field and a ["cc-exp-month"](#) field, and the user agent prefilled the form, then the month component of the former would have to match the latter. This requirement interacts with the [autofill anchor mantle](#) also. Consider the following markup snippet:

```
<form>
  <input type=hidden autocomplete="nickname" value="TreePlate">
  <input type=text autocomplete="nickname">
</form>
```

The only value that a conforming user agent could suggest in the text control is "TreePlate", the value given by the hidden [input](#) element.

The `"section-"` tokens in the [autofill scope](#) are opaque; user agents must not attempt to derive meaning from the precise values of these tokens.

For example, it would not be conforming if the user agent decided that it should offer the address it knows to be the user's daughter's address for `"section-child"` and the addresses it knows to be the user's spouses' addresses for `"section-spouse"`.

The autocompletion mechanism must be implemented by the user agent acting as if the user had modified the [control's data](#), and must be done at a time where the element is [mutable](#) (e.g. just after the element has been inserted into the document,



or when the user agent [stops parsing](#)). User agents must only prefill controls using values that the user could have entered.

For example, if a [select](#) element only has [option](#) elements with values "Steve" and "Rebecca", "Jay", and "Bob", and has an [autofill field name](#) "[given-name](#)", but the user agent's only idea for what to prefill the field with is "Evan", then the user agent cannot prefill the field. It would not be conforming to somehow set the [select](#) element to the value "Evan", since the user could not have done so themselves.

A user agent prefilling a form control must not discriminate between form controls that are [in a document tree](#) and those that are [connected](#); that is, it is not conforming to make the decision on whether or not to autofill based on whether the element's [root](#) is a [shadow root](#) versus a [Document](#).

A user agent prefilling a form control's [value](#) must not cause that control to [suffer from a type mismatch](#), [suffer from being too long](#), [suffer from being too short](#), [suffer from an underflow](#), [suffer from an overflow](#), or [suffer from a step mismatch](#). A user agent prefilling a form control's [value](#) must not cause that control to [suffer from a pattern mismatch](#) either. Where possible given the control's constraints, user agents must use the format given as canonical in the aforementioned table. Where it's not possible for the canonical format to be used, user agents should use heuristics to attempt to convert values so that they can be used.

For example, if the user agent knows that the user's middle name is "Ines", and attempts to prefill a form control that looks like this:

```
<input name=middle-initial maxlength=1 autocomplete="additional-name">
```

...then the user agent could convert "Ines" to "I" and prefill it that way.

A more elaborate example would be with month values. If the user agent knows that the user's birthday is the 27th of July 2012, then it might try to prefill all of the following controls with slightly different values, all driven from this information:

<pre>&lt;input name=b type=month autocomplete="bday"&gt;</pre>	2012-07	The day is dropped since the <a href="#">Month</a> state only accepts a month/year combination. (Note that this example is non-conforming, because the <a href="#">autofill field name</a> <a href="#">bday</a> is not allowed with the <a href="#">Month</a> state.)
<pre>&lt;select name=c autocomplete="bday"&gt;   &lt;option&gt;Jan   &lt;option&gt;Feb   ...   &lt;option&gt;Jul   &lt;option&gt;Aug</pre>	July	The user agent picks the month from the listed options, either by noticing there are twelve options and picking the 7th, or by recognizing that one of the strings (three characters "Jul" followed by a newline and a space) is a close match for the name of the month (July) in one of the user agent's supported languages, or through some other similar mechanism.

<pre>... &lt;/select&gt;</pre>		
<pre>&lt;input name=a type=number min=1 max=12 autocomplete="bday- month"&gt;</pre>	7	User agent converts "July" to a month number in the range 1..12, like the field.
<pre>&lt;input name=a type=number min=0 max=11 autocomplete="bday- month"&gt;</pre>	6	User agent converts "July" to a month number in the range 0..11, like the field.
<pre>&lt;input name=a type=number min=1 max=11 autocomplete="bday- month"&gt;</pre>		User agent doesn't fill in the field, since it can't make a good guess as to what the form expects.

A user agent may allow the user to override an element's [autofill field name](#), e.g. to change it from "[off](#)" to "[on](#)" to allow values to be remembered and prefilled despite the page author's objections, or to always "[off](#)", never remembering values.

More specifically, user agents may in particular consider replacing the [autofill field name](#) of form controls that match the description given in the first column of the following table, when their [autofill field name](#) is either "[on](#)" or "[off](#)", with the value given in the second cell of that row. If this table is used, the replacements must be done in [tree order](#), since all but the first row references the [autofill field name](#) of earlier elements. When the descriptions below refer to form controls being preceded or followed by others, they mean in the list of [listed elements](#) that share the same [form owner](#).

Form control	New <a href="#">autofill field name</a>
an <a href="#">input</a> element whose <a href="#">type</a> attribute is in the <a href="#">Text</a> state that is followed by an <a href="#">input</a> element whose <a href="#">type</a> attribute is in the <a href="#">Password</a> state	" <a href="#">username</a> "
an <a href="#">input</a> element whose <a href="#">type</a> attribute is in the <a href="#">Password</a> state that is preceded by an <a href="#">input</a> element whose <a href="#">autofill field name</a> is " <a href="#">username</a> "	" <a href="#">current-password</a> "
an <a href="#">input</a> element whose <a href="#">type</a> attribute is in the <a href="#">Password</a> state that is preceded by an <a href="#">input</a> element whose <a href="#">autofill field name</a> is " <a href="#">current-password</a> "	" <a href="#">new-password</a> "
an <a href="#">input</a> element whose <a href="#">type</a> attribute is in the <a href="#">Password</a> state that is preceded by an <a href="#">input</a> element whose <a href="#">autofill field name</a> is " <a href="#">new-password</a> "	" <a href="#">new-password</a> "

The [autocomplete](#) IDL attribute, on getting, must return the element's [IDL-exposed autofill value](#), and on setting, must [reflect](#) the content attribute of the same name.

#### 4.10.19 APIs for the text control selections

The `input` and `textarea` elements define several attributes and methods for handling their selection. Their shared algorithms are defined here.

`element.select()`

Selects everything in the text control.

`element.selectionStart [ = value ]`

Returns the offset to the start of the selection.

Can be set, to change the start of the selection.

`element.selectionEnd [ = value ]`

Returns the offset to the end of the selection.

Can be set, to change the end of the selection.

`element.selectionDirection [ = value ]`

Returns the current direction of the selection.

Can be set, to change the direction of the selection.

The possible values are "forward", "backward", and "none".

`element.setSelectionRange(start, end [, direction])`

✓MDN

Changes the selection to cover the given substring in the given direction. If the direction is omitted, it will be reset to be the platform default (none or forward).

`element.setRangeText(replacement [, start, end [, selectionMode ] ])`

✓MDN

Replaces a range of text with the new text. If the *start* and *end* arguments are not provided, the range is assumed to be the selection.

The final argument determines how the selection will be set after the text has been replaced. The possible values are:

**"select"**

Selects the newly inserted text.

**"start"**

Moves the selection to just before the inserted text.

**"end"**

Moves the selection to just after the selected text.

**"preserve"**

Attempts to preserve the selection. This is the default.

All `input` elements to which these APIs [apply](#), and all `textarea` elements, have either a **selection** or a **text entry cursor position** at all times (even for elements that are not [being rendered](#)), measured in offsets into the [code units](#) of the

control's [relevant value](#). The initial state must consist of a [text entry cursor](#) at the beginning of the control.

For [input](#) elements, these APIs must operate on the element's [value](#).

For [textarea](#) elements, these APIs must operate on the element's [API value](#). In the below algorithms, we call the value string being operated on the **relevant value**.

The use of [API value](#) instead of [raw value](#) for [textarea](#) elements means that U+000D (CR) characters are normalized away. For example,

```
<textarea id="demo"></textarea>
<script>
  demo.value = "A\r\nB";
  demo.setRangeText("replaced", 0, 2);
  assert(demo.value === "replacedB");
</script>
```

If we had operated on the [raw value](#) of "A\r\nB", then we would have replaced the characters "A\r", ending up with a result of "replaced\nB". But since we used the [API value](#) of "A\nB", we replaced the characters "A\n", giving "replacedB".

*Characters with no visible rendering, such as U+200D ZERO WIDTH JOINER, still count as characters. Thus, for instance, the selection can include just an invisible character, and the text insertion cursor can be placed to one side or another of such a character.*

Whenever the [relevant value](#) changes for an element to which these APIs apply, run these steps:

1. If the element has a [selection](#):
  1. If the start of the selection is now past the end of the [relevant value](#), set it to the end of the [relevant value](#).
  2. If the end of the selection is now past the end of the [relevant value](#), set it to the end of the [relevant value](#).
  3. If the user agent does not support empty selection, and both the start and end of the selection are now pointing to the end of the [relevant value](#), then instead set the element's [text entry cursor position](#) to the end of the [relevant value](#), removing any selection.
2. Otherwise, the element must have a [text entry cursor position](#) position. If it is now past the end of the [relevant value](#), set it to the end of the [relevant value](#).

*In some cases where the [relevant value](#) changes, other parts of the specification will also modify the [text entry cursor position](#), beyond just the clamping steps above. For example, see the [value setter for textarea](#).*

Where possible, user interface features for changing the [text selection](#) in [input](#) and [textarea](#) elements must be implemented using the [set the selection range](#) algorithm so that, e.g., all the same events fire.

The [selections](#) of [input](#) and [textarea](#) elements have a **selection direction**, which is either "forward", "backward", or "none". The exact meaning of the selection direction depends on the platform. This direction is set when the user manipulates the selection. The initial [selection direction](#) must be "none" if the platform supports that direction, or "forward" otherwise.

To **set the selection direction** of an element to a given direction, update the element's [selection direction](#) to the given direction, unless the direction is "none" and the platform does not support that direction; in that case, update the element's [selection direction](#) to "forward".

*On Windows, the direction indicates the position of the caret relative to the selection: a "forward" selection has the caret at the end of the selection and a "backward" selection has the caret at the start of the selection. Windows has no "none" direction.*

*On Mac, the direction indicates which end of the selection is affected when the user adjusts the size of the selection using the arrow keys with the Shift modifier: the "forward" direction means the end of the selection is modified, and the "backward" direction means the start of the selection is modified. The "none" direction is the default on Mac, it indicates that no particular direction has yet been selected. The user sets the direction implicitly when first adjusting the selection, based on which directional arrow key was used.*



The `select()` method, when invoked, must run the following steps:

1. If this element is an [input](#) element, and either [select\(\)](#) [does not apply](#) to this element or the corresponding control has no selectable text, return.

For instance, in a user agent where `<input type=color>` is rendered as a color well with a picker, as opposed to a text control accepting a hexadecimal color code, there would be no selectable text, and thus calls to the method are ignored.

2. [Set the selection range](#) with 0 and infinity.

The `selectionStart` attribute's getter must run the following steps:

1. If this element is an [input](#) element, and [selectionStart](#) [does not apply](#) to this element, return null.
2. If there is no [selection](#), return the [code unit](#) offset within the [relevant value](#) to the character that immediately follows the [text entry cursor](#).

3. Return the [code unit](#) offset within the [relevant value](#) to the character that immediately follows the start of the [selection](#).

The [selectionStart](#) attribute's setter must run the following steps:

1. If this element is an [input](#) element, and [selectionStart](#) [does not apply](#) to this element, throw an ["InvalidStateError"](#) [DOMException](#).
2. Let *end* be the value of this element's [selectionEnd](#) attribute.
3. If *end* is less than the given value, set *end* to the given value.
4. [Set the selection range](#) with the given value, *end*, and the value of this element's [selectionDirection](#) attribute.

The [selectionEnd](#) attribute's getter must run the following steps:

1. If this element is an [input](#) element, and [selectionEnd](#) [does not apply](#) to this element, return null.
2. If there is no [selection](#), return the [code unit](#) offset within the [relevant value](#) to the character that immediately follows the [text entry cursor](#).
3. Return the [code unit](#) offset within the [relevant value](#) to the character that immediately follows the end of the [selection](#).

The [selectionEnd](#) attribute's setter must run the following steps:

1. If this element is an [input](#) element, and [selectionEnd](#) [does not apply](#) to this element, throw an ["InvalidStateError"](#) [DOMException](#).
2. [Set the selection range](#) with the value of this element's [selectionStart](#) attribute, the given value, and the value of this element's [selectionDirection](#) attribute.

The [selectionDirection](#) attribute's getter must run the following steps:

1. If this element is an [input](#) element, and [selectionDirection](#) [does not apply](#) to this element, return null.
2. Return this element's [selection direction](#).

The [selectionDirection](#) attribute's setter must run the following steps:

1. If this element is an [input](#) element, and [selectionDirection](#) [does not apply](#) to this element, throw an ["InvalidStateError"](#) [DOMException](#).

2. [Set the selection range](#) with the value of this element's [selectionStart](#) attribute, the value of this element's [selectionEnd](#) attribute, and the given value.

The `setSelectionRange(start, end, direction)` method, when invoked, must run the following steps:

1. If this element is an `input` element, and `setSelectionRange()` [does not apply](#) to this element, throw an `"InvalidStateError" DOMException`.
2. [Set the selection range](#) with `start`, `end`, and `direction`.

To **set the selection range** with an integer or null `start`, an integer or null or the special value infinity `end`, and optionally a string `direction`, run the following steps:

1. If `start` is null, let `start` be zero.
2. If `end` is null, let `end` be zero.
3. Set the [selection](#) of the text control to the sequence of [code units](#) within the [relevant value](#) starting with the code unit at the `start`th position (in logical order) and ending with the code unit at the `(end-1)`th position. Arguments greater than the [length](#) of the [relevant value](#) of the text control (including the special value infinity) must be treated as pointing at the end of the text control. If `end` is less than or equal to `start` then the start of the selection and the end of the selection must both be placed immediately before the character with offset `end`. In UAs where there is no concept of an empty selection, this must set the cursor to be just before the character with offset `end`.
4. If `direction` is not [identical to](#) either `"backward"` or `"forward"`, or if the `direction` argument was not given, set `direction` to `"none"`.
5. [Set the selection direction](#) of the text control to `direction`.
6. If the previous steps caused the [selection](#) of the text control to be modified (in either extent or [direction](#)), then [queue an element task](#) on the [user interaction task source](#) given the element to [fire an event](#) named `select` at the element, with the [bubbles](#) attribute initialized to true.

The `setRangeText(replacement, start, end, selectMode)` method, when invoked, must run the following steps:

1. If this element is an `input` element, and `setRangeText()` [does not apply](#) to this element, throw an `"InvalidStateError" DOMException`.
2. Set this element's [dirty value flag](#) to true.
3. If the method has only one argument, then let `start` and `end` have the values of the [selectionStart](#) attribute and the [selectionEnd](#) attribute respectively.



Otherwise, let *start*, *end* have the values of the second and third arguments respectively.

4. If *start* is greater than *end*, then throw an `"IndexSizeError"` `DOMException`.
5. If *start* is greater than the `length` of the `relevant value` of the text control, then set it to the `length` of the `relevant value` of the text control.
6. If *end* is greater than the `length` of the `relevant value` of the text control, then set it to the `length` of the `relevant value` of the text control.
7. Let *selection start* be the current value of the `selectionStart` attribute.
8. Let *selection end* be the current value of the `selectionEnd` attribute.
9. If *start* is less than *end*, delete the sequence of `code units` within the element's `relevant value` starting with the code unit at the *start*th position and ending with the code unit at the (*end*-1)th position.
10. Insert the value of the first argument into the text of the `relevant value` of the text control, immediately before the *start*th `code unit`.
11. Let *new length* be the `length` of the value of the first argument.
12. Let *new end* be the sum of *start* and *new length*.
13. Run the appropriate set of substeps from the following list:

**If the fourth argument's value is `"select"`**

Let *selection start* be *start*.

Let *selection end* be *new end*.

**If the fourth argument's value is `"start"`**

Let *selection start* and *selection end* be *start*.

**If the fourth argument's value is `"end"`**

Let *selection start* and *selection end* be *new end*.

**If the fourth argument's value is `"preserve"`**

**If the method has only one argument**

1. Let *old length* be *end* minus *start*.
2. Let *delta* be *new length* minus *old length*.
3. If *selection start* is greater than *end*, then increment it by *delta*.  
(If *delta* is negative, i.e. the new text is shorter than the old text, then this will *decrease* the value of *selection start*.)



Otherwise: if *selection start* is greater than *start*, then set it to *start*.  
(This snaps the start of the selection to the start of the new text if it was in the middle of the text that it replaced.)

4. If *selection end* is greater than *end*, then increment it by *delta* in the same way.

Otherwise: if *selection end* is greater than *start*, then set it to *new end*.  
(This snaps the end of the selection to the end of the new text if it was in the middle of the text that it replaced.)

14. [Set the selection range](#) with *selection start* and *selection end*.

The [setRangeText\(\)](#) method uses the following enumeration:

```
enum SelectionMode {  
    "select",  
    "start",  
    "end",  
    "preserve" // default  
};
```

---

To obtain the currently selected text, the following JavaScript suffices:

```
var selectionText = control.value.substring(control.selectionStart,  
control.selectionEnd);
```

...where *control* is the [input](#) or [textarea](#) element.

To add some text at the start of a text control, while maintaining the text selection, the three attributes must be preserved:

```
var oldStart = control.selectionStart;  
var oldEnd = control.selectionEnd;  
var oldDirection = control.selectionDirection;  
var prefix = "http://";  
control.value = prefix + control.value;
```

```
control.setSelectionRange(oldStart + prefix.length, oldEnd +  
prefix.length, oldDirection);
```

...where *control* is the [input](#) or [textarea](#) element.

## 4.10.20 Constraints

### 4.10.20.1 Definitions

A [submittable element](#) is a **candidate for constraint validation** except when a condition has **barred the element from constraint validation**. (For example, an element is [barred from constraint validation](#) if it has a [datalist](#) element ancestor.)

An element can have a **custom validity error message** defined. Initially, an element must have its [custom validity error message](#) set to the empty string. When its value is not the empty string, the element is [suffering from a custom error](#). It can be set using the [setCustomValidity\(\)](#) method, except for [form-associated custom elements](#). [Form-associated custom elements](#) can have a [custom validity error message](#) set via their [ElementInternals](#) object's [setValidity\(\)](#) method. The user agent should use the [custom validity error message](#) when alerting the user to the problem with the control.

An element can be constrained in various ways. The following is the list of **validity states** that a form control can be in, making the control invalid for the purposes of constraint validation. (The definitions below are non-normative; other parts of this specification define more precisely when each state applies or does not.)

#### ***Suffering from being missing***

When a control has no [value](#) but has a `required` attribute ([input required](#), [textarea required](#)); or, more complicated rules for [select](#) elements and controls in [radio button groups](#), as specified in their sections.

When the [setValidity\(\)](#) method sets `valueMissing` flag to true for a [form-associated custom element](#).

#### ***Suffering from a type mismatch***

When a control that allows arbitrary user input has a [value](#) that is not in the correct syntax ([Email](#), [URL](#)).

When the [setValidity\(\)](#) method sets `typeMismatch` flag to true for a [form-associated custom element](#).

#### ***Suffering from a pattern mismatch***

When a control has a [value](#) that doesn't satisfy the [pattern](#) attribute.

When the `setValidity()` method sets `patternMismatch` flag to true for a [form-associated custom element](#).

### ***Suffering from being too long***

When a control has a [value](#) that is too long for the [form control](#) `maxlength` attribute (`input maxlength`, `textarea maxlength`).

When the `setValidity()` method sets `tooLong` flag to true for a [form-associated custom element](#).

### ***Suffering from being too short***

When a control has a [value](#) that is too short for the [form control](#) `minlength` attribute (`input minlength`, `textarea minlength`).

When the `setValidity()` method sets `tooShort` flag to true for a [form-associated custom element](#).

### ***Suffering from an underflow***

When a control has a [value](#) that is not the empty string and is too low for the `min` attribute.

When the `setValidity()` method sets `rangeUnderflow` flag to true for a [form-associated custom element](#).

### ***Suffering from an overflow***

When a control has a [value](#) that is not the empty string and is too high for the `max` attribute.

When the `setValidity()` method sets `rangeOverflow` flag to true for a [form-associated custom element](#).

### ***Suffering from a step mismatch***

When a control has a [value](#) that doesn't fit the rules given by the `step` attribute.

When the `setValidity()` method sets `stepMismatch` flag to true for a [form-associated custom element](#).

### ***Suffering from bad input***

When a control has incomplete input and the user agent does not think the user ought to be able to submit the form in its current state.

When the `setValidity()` method sets `badInput` flag to true for a [form-associated custom element](#).

### ***Suffering from a custom error***

When a control's [custom validity error message](#) (as set by the element's `setCustomValidity()` method or `ElementInternals`'s `setValidity()` method) is not the empty string.

*An element can still suffer from these states even when the element is [disabled](#); thus these states can be represented in the DOM even if validating the form during submission wouldn't indicate a problem to the user.*

An element **satisfies its constraints** if it is not suffering from any of the above [validity states](#).

#### 4.10.20.2 Constraint validation

When the user agent is required to **statically validate the constraints** of [form](#) element *form*, it must run the following steps, which return either a *positive* result (all the controls in the form are valid) or a *negative* result (there are invalid controls) along with a (possibly empty) list of elements that are invalid and for which no script has claimed responsibility:

1. Let *controls* be a list of all the [submittable elements](#) whose [form owner](#) is *form*, in [tree order](#).
2. Let *invalid controls* be an initially empty list of elements.
3. For each element *field* in *controls*, in [tree order](#):
  1. If *field* is not a [candidate for constraint validation](#), then move on to the next element.
  2. Otherwise, if *field* [satisfies its constraints](#), then move on to the next element.
  3. Otherwise, add *field* to *invalid controls*.
4. If *invalid controls* is empty, then return a *positive* result.
5. Let *unhandled invalid controls* be an initially empty list of elements.
6. For each element *field* in *invalid controls*, if any, in [tree order](#):
  1. Let *notCanceled* be the result of [firing an event](#) named [invalid](#) at *field*, with the [cancelable](#) attribute initialized to true.
  2. If *notCanceled* is true, then add *field* to *unhandled invalid controls*.
7. Return a *negative* result with the list of elements in the *unhandled invalid controls* list.

If a user agent is to **interactively validate the constraints** of form element *form*, then the user agent must run the following steps:

1. [Statically validate the constraints](#) of *form*, and let *unhandled invalid controls* be the list of elements returned if the result was *negative*.
2. If the result was *positive*, then return that result.
3. Report the problems with the constraints of at least one of the elements given in *unhandled invalid controls* to the user.
  - User agents may focus one of those elements in the process, by running the [focusing steps](#) for that element, and may change the scrolling position of the document, or perform some other action that brings the element to the user's attention. For elements that are [form-associated custom elements](#), user agents should use their [validation anchor](#) instead, for the purposes of these actions.
  - User agents may report more than one constraint violation.
  - User agents may coalesce related constraint violation reports if appropriate (e.g. if multiple radio buttons in a [group](#) are marked as required, only one error need be reported).
  - If one of the controls is not [being rendered](#) (e.g. it has the hidden attribute set) then user agents may report a script error.
4. Return a *negative* result.

#### 4.10.20.3 The *constraint validation API*

`element.willValidate`



Returns true if the element will be validated when the form is submitted; false otherwise.

`element.setCustomValidity(message)`



Sets a custom error, so that the element would fail to validate. The given message is the message to be shown to the user when reporting the problem to the user.

If the argument is the empty string, clears the custom error.

`element.validity.valueMissing`

Returns true if the element has no value but is a required field; false otherwise.

`element.validity.typeMismatch`

Returns true if the element's value is not in the correct syntax; false otherwise.

`element.validity.patternMismatch`

Returns true if the element's value doesn't match the provided pattern; false otherwise.

`element.validity.tooLong`

✓MDN

Returns true if the element's value is longer than the provided maximum length; false otherwise.

`element.validity.tooShort`

✓MDN

Returns true if the element's value, if it is not the empty string, is shorter than the provided minimum length; false otherwise.

`element.validity.rangeUnderflow`

Returns true if the element's value is lower than the provided minimum; false otherwise.

`element.validity.rangeOverflow`

Returns true if the element's value is higher than the provided maximum; false otherwise.

`element.validity.stepMismatch`

Returns true if the element's value doesn't fit the rules given by the `step` attribute; false otherwise.

`element.validity.badInput`

✓MDN

Returns true if the user has provided input in the user interface that the user agent is unable to convert to a value; false otherwise.

`element.validity.customError`

Returns true if the element has a custom error; false otherwise.

`element.validity.valid`

Returns true if the element's value has no validity problems; false otherwise.

`valid = element.checkValidity()`

✓MDN

Returns true if the element's value has no validity problems; false otherwise. Fires an `invalid` event at the element in the latter case.

`valid = element.reportValidity()`

✓MDN

Returns true if the element's value has no validity problems; otherwise, returns false, fires an [invalid](#) event at the element, and (if the event isn't canceled) reports the problem to the user.

#### `element.validationMessage`



Returns the error message that would be shown to the user if the element was to be checked for validity.

The `willValidate` attribute's getter must return true, if this element is a [candidate for constraint validation](#), and false otherwise (i.e., false if any conditions are [barring it from constraint validation](#)).



The `willValidate` attribute of [ElementInternals](#) interface, on getting, must throw a `"NotSupportedError" DOMException` if the [target element](#) is not a [form-associated custom element](#). Otherwise, it must return true if the [target element](#) is a [candidate for constraint validation](#), and false otherwise.



The `setCustomValidity(error)` method, when invoked, must set the [custom validity error message](#) to `error`.

In the following example, a script checks the value of a form control each time it is edited, and whenever it is not a valid value, uses the [setCustomValidity\(\)](#) method to set an appropriate message.

```
<label>Feeling: <input name=f type="text"
oninput="check(this)"></label>

<script>
  function check(input) {
    if (input.value == "good" ||
        input.value == "fine" ||
        input.value == "tired") {
      input.setCustomValidity('"' + input.value + '" is not a
feeling. ');
    } else {
      // input is fine -- reset the error message
      input.setCustomValidity('');
    }
  }
</script>
```



The **validity** attribute's getter must return a [ValidityState](#) object that represents the [validity states](#) of this element. This object is [live](#).

## MDN

L' **validity** attribut de [ElementInternals](#) l'interface, lors de l'obtention, doit lancer un " [NotSupportedError](#) " [DOMException](#) si l' [élément cible](#) n'est pas un [élément personnalisé associé au formulaire](#) . Sinon, il doit retourner un [ValidityState](#) objet qui représente les [états de validité](#) de l' [élément cible](#) . Cet objet est [en direct](#) .

```
[Exposed=Window]

interface ValidityState {

    readonly attribute boolean valueMissing;

    readonly attribute boolean typeMismatch;

    readonly attribute boolean patternMismatch;

    readonly attribute boolean tooLong;

    readonly attribute boolean tooShort;

    readonly attribute boolean rangeUnderflow;

    readonly attribute boolean rangeOverflow;

    readonly attribute boolean stepMismatch;

    readonly attribute boolean badInput;

    readonly attribute boolean customError;

    readonly attribute boolean valid;

};
```

Un [ValidityState](#) objet a les attributs suivants. A l'obtention, elles doivent retourner true si la condition correspondante donnée dans la liste suivante est true, et false sinon.

### **valueMissing**

Le contrôle [souffre d'être manquant](#) .



#### typeMismatch



Le contrôle [souffre d'une incompatibilité de type](#) .

#### patternMismatch



Le contrôle [souffre d'une incompatibilité de modèle](#) .

#### tooLong

Le contrôle [souffre d'être trop long](#) .

#### tooShort

Le contrôle [souffre d'être trop court](#) .

#### rangeUnderflow



Le contrôle [souffre d'un débordement insuffisant](#) .

#### rangeOverflow



Le contrôle [souffre d'un débordement](#) .

#### stepMismatch



Le contrôle [souffre d'une non-concordance d'étape](#) .

#### badInput

Le contrôle [souffre d'une mauvaise entrée](#) .

#### customError

Le contrôle [souffre d'une erreur personnalisée](#) .

#### valid

Aucune des autres conditions n'est vraie.

Les **étapes de vérification de la validité** d'un *élément* element sont :

1. Si l'*élément* est [candidat à la validation de contrainte](#) et ne [satisfait pas ses contraintes](#) , alors :
  1. [Lancez un événement](#) nommé `invalid` à *element* , avec l' `cancelable` attribut initialisé à true (bien que l'annulation n'ait aucun effet).
  2. Renvoie faux.

2. Renvoie vrai.

La `checkValidity()` méthode, lorsqu'elle est invoquée, doit exécuter les [étapes de vérification de validité](#) sur cet élément.

MDN

La `checkValidity()` méthode de l' `ElementInternals` interface doit exécuter ces étapes :

1. Soit *element* [l'élément cible](#) `ElementInternals` de this .
2. Si *l'élément* n'est pas un [élément personnalisé associé au formulaire](#) , lancez un `"NotSupportedError"` `DOMException` .
3. Exécutez les [étapes de vérification de la validité](#) sur *l'élément* .

Les **étapes de validité du rapport** pour un *élément* sont :

1. Si *l'élément* est [candidat à la validation de contrainte](#) et ne [satisfait pas ses contraintes](#) , alors :
  1. Soit *report* le résultat du [déclenchement d'un événement](#) nommé `invalid` à *element* , avec l' `cancelable` attribut initialisé à true.
  2. Si *report* est vrai, alors signalez les problèmes avec les contraintes de cet élément à l'utilisateur. Lorsqu'il signale le problème avec les contraintes à l'utilisateur, l'agent utilisateur peut exécuter les [étapes de focalisation](#) pour *l'élément* , et peut changer la position de défilement du document, ou effectuer une autre action qui attire l'attention de l'utilisateur sur l' *élément* . Les agents utilisateurs peuvent signaler plus d'une violation de contrainte, si *l'élément* souffre de plusieurs problèmes à la fois. Si *l'élément* n'est pas [rendu](#) , alors l'agent utilisateur peut, au lieu de notifier l'utilisateur, [signaler l'erreur](#) pour le [script en cours d'exécution](#).
  3. Renvoie faux.
2. Renvoie vrai.

La `reportValidity()` méthode, lorsqu'elle est invoquée, doit exécuter les [étapes de validité du rapport](#) sur cet élément.

MDN

La `reportValidity()` méthode de l' `ElementInternals` interface doit exécuter ces étapes :

1. Soit *element* [l'élément cible](#)`ElementInternals` de this .
2. Si *l'élément* n'est pas un [élément personnalisé associé au formulaire](#) , lancez un `"NotSupportedError"` `DOMException` .
3. Exécutez les [étapes de validité du rapport](#) sur *l'élément* .

Le `validationMessage` getter de l'attribut doit exécuter ces étapes :

1. Si cet élément n'est pas un [candidat pour la validation de contrainte](#) ou si cet élément [satisfait ses contraintes](#) , alors retournez la chaîne vide.
2. Renvoie un message convenablement localisé que l'agent utilisateur montrerait à l'utilisateur s'il s'agissait du seul contrôle de formulaire avec un problème de contrainte de validité. Si l'agent utilisateur n'afficherait pas réellement un message textuel dans une telle situation (par exemple, il afficherait un repère graphique à la place), alors renvoyez un message convenablement localisé qui exprime (une ou plusieurs) la ou les contraintes de validité que le contrôle ne satisfait pas. Si l'élément est [candidat à la validation de contrainte](#) et [souffre d'une erreur personnalisée](#) , le [message d'erreur de validité personnalisé](#) doit être présent dans la valeur de retour.

#### 4.10.20.4 Sécurité

Les serveurs ne doivent pas compter sur la validation côté client. La validation côté client peut être intentionnellement contournée par des utilisateurs hostiles, et involontairement contournée par des utilisateurs d'agents utilisateurs plus anciens ou d'outils automatisés qui n'implémentent pas ces fonctionnalités. Les fonctionnalités de validation des contraintes sont uniquement destinées à améliorer l'expérience utilisateur, et non à fournir un quelconque mécanisme de sécurité.

### 4.10.21 Soumission du formulaire

#### 4.10.21.1 Présentation

*Cette section est non normative.*

Lorsqu'un formulaire est soumis, les données du formulaire sont converties dans la structure spécifiée par le [enctype](#) , puis envoyées à la destination spécifiée par l' [action à l'aide de la méthode](#) donnée .

Prenons par exemple la forme suivante :

```
<form action="/find.cgi" method=get>
```

```



```

Si l'utilisateur tape "cats" dans le premier champ et "fur" dans le second, puis appuie sur le bouton submit, l'agent utilisateur chargera `/find.cgi?t=cats&q=fur`.

D'autre part, considérez cette forme:

```

<form action="/find.cgi" method=post enctype="multipart/form-data">


```

Étant donné la même entrée utilisateur, le résultat lors de la soumission est assez différent : l'agent utilisateur effectue à la place un HTTP POST vers l'URL donnée, avec comme corps d'entité quelque chose comme le texte suivant :

```

-----kYFrd4jNJEgCervE
Content-Disposition : formulaire-données ; nom="t"

chats
-----kYFrd4jNJEgCervE
Content-Disposition : formulaire-données ; nom="q"

fourrure
-----kYFrd4jNJEgCervE--

```

#### 4.10.21.2 Soumission implicite

Le **bouton par défaut**[form](#) d'un élément est le premier [bouton d'envoi](#) dans [l'arborescence](#) dont [le propriétaire du formulaire](#) est cet élément.[form](#)

Si l'agent utilisateur prend en charge le fait de laisser l'utilisateur soumettre un formulaire implicitement (par exemple, sur certaines plates-formes, appuyer sur la touche "Entrée" alors qu'un contrôle de texte est focalisé soumet [implicitement](#) le formulaire), alors faites-le pour un formulaire, dont [le bouton par défaut](#) a [un comportement d'activation](#) et n'est pas [désactivé](#) , doit forcer l'agent utilisateur à [déclencher un click](#)[événement](#) sur ce [bouton par défaut](#) .

*Il existe des pages sur le Web qui ne sont utilisables que s'il existe un moyen de soumettre implicitement des formulaires, les agents utilisateurs sont donc fortement encouragés à le prendre en charge.*

Si le formulaire n'a pas [de bouton de soumission](#) , alors le mécanisme de soumission implicite ne doit rien faire si le formulaire a plus d'un *champ qui bloque la soumission implicite* et doit [soumettre](#) l' [form](#) élément à partir de l' [form](#)élément lui-même sinon.

Aux fins du paragraphe précédent, un élément est un *champ qui bloque la soumission implicite* d'un [form](#)élément s'il s'agit d'un [input](#)élément dont [le propriétaire du formulaire](#) est cet [form](#)élément et dont [type](#)l'attribut est dans l'un des états suivants : [Text](#) , [Search](#) , [URL](#) , [Telephone](#) , [Email](#) , [Mot de passe](#) , [Date](#) , [Mois](#) , [Semaine](#) , [Heure](#) , [Date et heure locales](#) , [Numéro](#)

#### 4.10.21.3 Algorithme de soumission de formulaire

Chaque [form](#)élément a une **liste d'entrées de construction** booléenne, initialement fausse.

Chaque [form](#)élément a un booléen **d'événements de soumission de déclenchement** , initialement faux.

Lorsqu'un *formulaire*[form](#) d'élément est **soumis** à partir d'un *émetteur* d'élément (généralement un bouton), éventuellement avec un indicateur de *méthode soumis* depuis défini, l'agent utilisateur doit exécuter les étapes suivantes :[submit\(\)](#)

1. Si le formulaire [ne peut pas naviguer](#) , alors revenez.
2. Si [la liste d'entrées de construction](#) du *formulaire* est vraie, alors retournez.
3. Soit *form document* le [nœud document](#) du *formulaire* .
4. Si [l'indicateur de bac à sable actif](#) du *document de formulaire* est défini , son indicateur [de contexte de navigation des formulaires en bac à sable](#) est défini, puis retournez.
5. Si l' *indicateur de* [submit\(\)](#) *méthode* soumis depuis n'est pas défini, alors :
  1. Si [les événements de soumission de déclenchement](#) du *formulaire* sont vrais, alors retournez.
  2. Définissez [les événements de soumission de déclenchement](#) du *formulaire* sur true.
  3. Si l' [état de non-validation](#) de l'élément *émetteur* est faux, [validez de manière interactive les contraintes](#) de *formulaire* et examinez le

résultat. Si le résultat est négatif (c'est-à-dire que la validation des contraintes a conclu qu'il y avait des champs invalides et en a probablement informé l'utilisateur), alors :

1. Définissez [les événements de soumission de déclenchement](#) du *formulaire* sur *false*.
2. Retour.
4. Laissez *submitterButton* être null si l'émetteur est *form* . Sinon, laissez *submitterButton* être *submitter* .
5. Soit *shouldContinue* le résultat du [déclenchement d'un événement](#) nommé *submit* au *formulaire* en utilisant *SubmitEvent*, avec l' *submitter* attribut initialisé à *submitterButton* , l' *bubbles* attribut initialisé à *true* et l' *cancelable* attribut initialisé à *true*.
6. Définissez [les événements de soumission de déclenchement](#) du *formulaire* sur *false*.
7. Si *shouldContinue* est faux, alors retournez.
8. Si le *formulaire* [ne peut pas naviguer](#) , alors revenez.

*[Impossible de naviguer](#) est réexécuté, car l'envoi de l' *submit* événement aurait pu modifier le résultat.*

6. Soit *encoding* le résultat de [la sélection d'un encodage pour le formulaire](#) .
7. Soit *liste d'entrées* le résultat de [la construction de la liste d'entrées](#) avec *form* , *submitter* et *encoding* .
8. [Assert](#) : la *liste d'entrées* n'est pas nulle.
9. Si le *formulaire* [ne peut pas naviguer](#) , alors revenez.

*[Impossible de naviguer](#) est réexécuté car l'envoi de l' *formdata* événement lors de [la construction de la liste d'entrées](#) aurait pu modifier le résultat.*

10. Soit *action* l' [action](#) de l'élément émetteur .
11. Si *action* est la chaîne vide, laissez *action* être l' [URL](#) du *document de formulaire* .
12. [Analyser une URL](#) donnée *action* , par rapport au [document de nœud](#) de l'élément émetteur . Si cela échoue, revenez.
13. Soit l'*action analysée* l' [enregistrement d'URL résultant](#) .
14. Soit *schéma* le [schéma](#) de l'*action analysée* .

15. Soit *enctype* le [enctype](#) de l'élément *émetteur* .
16. Soit *method* la [méthode](#) de l'élément *émetteur* .
17. Soit *target* la valeur d'attribut de l'élément *émetteur* [formtarget](#) , si l'élément est un [bouton d'envoi](#) et possède un tel attribut. Sinon, supposons que ce soit le résultat de [l'obtention de la cible d'un élément](#) en fonction [du propriétaire](#) du formulaire de *l'émetteur* .
18. Soit *noopener* le résultat de [l'obtention du noopener d'un élément](#) avec *form* et *target* .
19. Soit *targetNavigable* la première valeur de retour de l'application [des règles de choix d'une cible](#) navigable donnée , d'un nœud de *formulaire* [navigable](#) et de *noopener* .
20. Si *targetNavigable* est null, alors retournez.
21. Soit *historyHandling* la valeur " [push](#)".
22. Si le document de formulaire n'est pas encore [complètement chargé](#) , définissez *historyHandling* sur " [replace](#)" .
23. Si la valeur de la *méthode* est [dialog](#) , passez aux étapes de la boîte [de dialogue de soumission](#) .

Sinon, sélectionnez la ligne appropriée dans le tableau ci-dessous en fonction de la valeur du *schéma* telle qu'elle est donnée par la première cellule de chaque ligne. Ensuite, sélectionnez la cellule appropriée sur cette ligne en fonction de la valeur de la *méthode* indiquée dans la première cellule de chaque colonne. Passez ensuite aux étapes nommées dans cette cellule et définies sous le tableau.

	<a href="#">OBTENIR</a>	<a href="#">POSTE</a>
http	<a href="#">Muter l'URL de l'action</a>	<a href="#">Soumettre en tant que corps d'entité</a>
https	<a href="#">Muter l'URL de l'action</a>	<a href="#">Soumettre en tant que corps d'entité</a>
ftp	<a href="#">Obtenir l'URL de l'action</a>	<a href="#">Obtenir l'URL de l'action</a>
javascript	<a href="#">Obtenir l'URL de l'action</a>	<a href="#">Obtenir l'URL de l'action</a>
data	<a href="#">Muter l'URL de l'action</a>	<a href="#">Obtenir l'URL de l'action</a>
mailto	<a href="#">Courrier avec en-têtes</a>	<a href="#">Courrier en tant que corps</a>

Si le *schéma* ne fait pas partie de ceux répertoriés dans ce tableau, le comportement n'est pas défini par cette spécification. Les agents utilisateurs devraient, en l'absence d'une autre spécification définissant cela, agir d'une manière analogue à celle définie dans cette spécification pour des schémas similaires.

Chaque [form](#)élément a une **navigation planifiée** , qui est soit nulle soit une [tâche](#) ; lorsque le [form](#)est créé pour la première fois, sa [navigation](#)

[planifiée](#) doit être définie sur null. Dans les comportements décrits ci-dessous, lorsque l'agent utilisateur doit **prévoir de naviguer** vers une [URL donnée avec une ressource POST](#) optionnelle -ou-null *postResource* (null par défaut), il doit exécuter les étapes suivantes :

1. Soit *referrerPolicy* la chaîne vide.
2. Si les [types de liens](#)*form* de l'élément incluent le mot-clé, définissez *referrerPolicy* sur " ".[noreferrer](#)*no-referrer*
3. Si le a une [navigation planifiée](#)*form* non nulle , supprimez-la de sa [file d'attente de tâches](#) .
4. [Mettez en file d'attente une tâche d'élément](#) sur la [source de la tâche de manipulation DOM](#) en fonction de l' *form*élément et des étapes suivantes :
  1. Définissez la [navigation planifiée](#)*form* de sur null.
  2. [Accédez](#) à *targetNavigable* vers *url* à l'aide du [nœud document](#)*form* de l'élément , avec [historyHandling](#) défini sur *historyHandling* , [referrerPolicy](#) défini sur *referrerPolicy* , [documentResource](#) défini sur *postResource* et [cspNavigationType](#) défini sur " ".*form-submission*
5. Définissez la [navigation planifiée](#)*form* de sur la [tâche](#) qui vient d'être mise en file d'attente .

Les comportements sont les suivants :

### **Muter l'URL de l'action**

Soit *les paires* le résultat de [la conversion en une liste de paires nom-valeur](#) avec l'entrée *list* .

Soit *query* le résultat de l'exécution du [application/x-www-form-urlencoded](#)*sérialiseur* avec *des paires* et *encoding* .

Définit le composant [de requête](#) de l'action analysée sur *query* .

[Prévoyez d'accéder](#) à l'action analysée .

### **Soumettre en tant que corps d'entité**

[Assert](#) : la méthode est [POST](#) .

Activer *enctype* :

### **[application/x-www-form-urlencoded](#)**

Soit *les paires* le résultat de [la conversion en une liste de paires nom-valeur](#) avec l'entrée *list* .



Soit *body* le résultat de l'exécution du [application/x-www-form-urlencoded](#)[séréaliseur](#) avec *des paires* et *encoding* .

Définissez *body* sur le résultat de [l'encodage](#) *body* .

Soit *mimeType* être ` [application/x-www-form-urlencoded](#) `.

### [multipart/form-data](#)

Soit *body* le résultat de l'exécution de [multipart/form-data](#)[l'algorithme d'encodage](#) avec *entry list* et *encoding* .

Soit *mimeType* le [codage isomorphe](#) de la concaténation de " multipart/form-data; boundary=" et de la [multipart/form-data](#)[chaîne limite](#) générée par l' [multipart/form-data](#)[algorithme de codage](#) .

### [text/plain](#)

Soit *les paires* le résultat de [la conversion en une liste de paires nom-valeur](#) avec *l'entrée list* .

Soit *body* le résultat de l'exécution de l' [text/plain](#) [algorithme d'encodage](#) avec *des paires* .

Définissez *body* sur le résultat de [l'encodage](#) *body* en utilisant *encoding* .

Soit *mimeType* être ` [text/plain](#) `.

[Prévoyez d'accéder](#) à *l'action analysée* en fonction d'une [ressource POST](#) dont [le corps de la demande](#) est *body* et [le type de contenu de la demande](#) est *mimeType* .

### **Obtenir l'URL de l'action**

[Prévoyez d'accéder](#) à *l'action analysée* .

*la liste des entrées est supprimée.*

### **Courrier avec en-têtes**

Soit *les paires* le résultat de [la conversion en une liste de paires nom-valeur](#) avec *l'entrée list* .

Laissez *les en-têtes* être le résultat de l'exécution du [application/x-www-form-urlencoded](#)[séréaliseur](#) avec *des paires* et *encoding* .

Remplacez les occurrences des caractères U+002B PLUS SIGN (+) dans *les en-têtes* par la chaîne " %20".

Définit [la requête](#) de *l'action analysée* sur *les en-têtes* .

[Prévoyez d'accéder](#) à *l'action analysée* .

### **Courrier en tant que corps**

Soit *les paires* le résultat de [la conversion en une liste de paires nom-valeur](#) avec l'entrée *list* .

Activer *enctype* :

#### **text/plain**

Soit *body* le résultat de l'exécution de l' [text/plain](#) *algorithme d'encodage* avec *des paires* .

Définissez *body* sur le résultat de l'exécution [de l'encodage en pourcentage UTF-8](#) sur *body* à l'aide du [jeu d'encodage par défaut](#) . [\[URL\]](#)

### **Sinon**

Soit *body* le résultat de l'exécution du [application/x-www-form-urlencoded](#) *sérialiseur* avec *des paires* et *encoding* .

Si [la requête](#) de l'action analysée est nulle, définissez-la sur la chaîne vide.

Si [la requête](#) de l'action analysée n'est pas la chaîne vide, ajoutez-lui un seul caractère U+0026 AMPERSAND (&).

Ajouter " *body*=" à [la requête](#) de l' *action analysée* .

Ajouter *le corps* à [la requête](#) de l'action analysée .

[Prévoyez d'accéder](#) à l'action analysée .

### **Boîte de dialogue Soumettre**

Soit *sujet* l'élément ancêtre le plus proche [dialog](#) de *form* , le cas échéant.

S'il n'y en a pas, ou s'il n'a pas d' [open](#) attribut, ne faites rien. Sinon, procédez comme suit :

Si l'émetteur est un [input](#) élément dont [type](#) l'attribut est dans l' état [Image Button](#) , alors laissez *result* être la chaîne formée en concaténant le composant *x* de [la coordonnée sélectionnée](#) , exprimé sous la forme d'un nombre en base dix à l'aide [de chiffres ASCII](#) , un caractère U+002C COMMA ( , ), et la composante *y* de [la coordonnée sélectionnée](#) , exprimée de la même manière que la composante *x* .

Sinon, si l'émetteur a une [valeur](#) , alors laissez *le résultat* être cette [valeur](#) .

Sinon, il n'y a pas *de résultat* .

Ensuite, [fermez le](#) *sujet* de la boîte de dialogue . S'il y a un *result* , que ce soit la valeur de retour.

#### 4.10.21.4 Construire la liste des inscrits

Une **liste d'entrées** est une liste d' entrées , représentant généralement le contenu d'un formulaire. Une **entrée** est un tuple composé d'un **nom** (une chaîne de valeur scalaire ) et d'une **valeur** (soit une chaîne de valeur scalaire , soit un Fileobjet).

Pour **créer une entrée** avec un *nom* de chaîne , une *valeur*Blob de chaîne ou d'objet , et éventuellement un *nom de fichier* de chaîne de valeur scalaire :

1. Définissez *name* sur le résultat de la conversion de *name* en une chaîne de valeur scalaire .
2. Si *value* est une chaîne, définissez *value* sur le résultat de la conversion de *value* en chaîne de valeur scalaire .
3. Sinon:
  1. Si *la valeur* n'est pas un Fileobjet, alors définissez *la valeur* sur un nouvel Fileobjet, représentant les mêmes octets, dont name la valeur d'attribut est " `blob` ".
  2. Si *filename* est donné, définissez *la valeur* sur un nouvel File objet, représentant les mêmes octets, dont name l'attribut est *filename* .

*Ces opérations créeront un nouvel Fileobjet si le nom de fichier est donné ou si le passé Blob n'est pas un Fileobjet. Dans ces cas, l'identité de l'Blob objet passé n'est pas conservée.*

4. Renvoie une entrée dont le nom est *name* et dont la valeur est *value* .

L'algorithme pour **construire la liste d'entrées** étant donné un *formulaire* , un *émetteur* facultatif et un *encodage* facultatif , est le suivant. Sauf indication contraire, l'*émetteur* est nul.

1. Si la construction de la liste d'entrées du *formulaire* est vraie, alors renvoie null.
2. Définissez la liste d'entrées de construction du *formulaire* sur true.
3. Soit *les contrôles* une liste de tous les éléments soumis dont le propriétaire du formulaire est *form* , dans l'ordre de l'arborescence .
4. Soit *liste d'entrées* une nouvelle liste d'entrées vide .
5. Pour chaque *champ* élément des *champs* , dans l'ordre de l'arborescence :
  1. Si l'une des conditions suivantes est vraie :
    - L'élément *champ* a un datalist ancêtre d'élément.
    - L'élément *champ* est désactivé .

- L'élément *field* est un [bouton](#) mais ce n'est pas un *émetteur* .
- L'élément *field* est un [input](#)élément dont [type](#)l'attribut est à l'état [Checkbox](#) et dont [la case à cocher](#) est false.
- L'élément *champ* est un [input](#)élément dont [type](#)l'attribut est à l'état [Bouton radio](#) et dont [la case à cocher](#) est fausse.

[Continuez](#) ensuite .

2. Si l'élément *field* est un [input](#)élément dont [type](#)l'attribut est à l'état [Image Button](#) , alors :
  - Si l'élément *de champ* n'est pas *émetteur* , [continuez](#) .
  - Si l'élément *de champ* a un [name](#) attribut spécifié et que sa valeur n'est pas la chaîne vide, *nommez* cette valeur suivie de U+002E (.). Sinon, laissez *name* être la chaîne vide.
  - Soit *nom<sub>x</sub>* la concaténation de *nom* et U+0078 (x).
  - Soit *nom<sub>y</sub>* la concaténation de *nom* et U+0079 (y).
  - Soit ( *x* , *y* ) la [coordonnée sélectionnée](#) .
  - [Créez une entrée](#) avec le *nom<sub>x</sub>* et *x* , et [ajoutez](#) -la à la liste d'entrées .
  - [Créez une entrée](#) avec le *nom<sub>y</sub>* et *y* , et [ajoutez](#) -la à la liste d'entrées .
  - [Continuez](#) .
3. Si le *champ* est un [élément personnalisé associé à un formulaire](#) , exécutez l' [algorithme de construction d'entrée](#) en fonction du *champ* et de la liste d'entrées , puis [continuez](#) .
4. Si l'élément *de champ* n'a pas d' [name](#)attribut spécifié ou si [name](#)la valeur de son attribut est une chaîne vide, [continuez](#) .
5. Soit *name* la valeur de l' attribut de l'élément *field*[name](#) .
6. Si l'élément *field* est un [select](#)élément, alors pour chaque [option](#)élément de la [liste d'options](#)[select](#) de l'élément dont [la sélection](#) est true et qui n'est pas [disabled](#) , [créez une entrée](#) avec le *nom* et la [valeur](#) de l' élément et [ajoutez](#) -la à *entry list* [option](#)
7. Sinon, si l'élément *field*[input](#) est un élément dont [type](#)l'attribut est à l'état [Checkbox](#) ou à l'état [Radio Button](#) , alors :

- Si l'élément *de champ* a un valueattribut spécifié, alors laissez *value* être la valeur de cet attribut ; sinon, laissez *value* être la chaîne " on".
  - Créez une entrée avec *un nom* et *une valeur* , et ajoutez - la à la *liste des entrées* .
8. Sinon, si l'élément *field* est un inputélément dont typel'attribut est à l'état File Upload , alors :
- S'il n'y a pas de fichiers sélectionnés , créez une entrée avec *un nom* et un nouvel Fileobjet avec un nom, application/octet-streamun type et un corps vides, puis ajoutez -le à la *liste des entrées* .
  - Sinon, pour chaque fichier dans les fichiers sélectionnés , créez une entrée avec *un nom* et un File objet représentant le fichier, et ajoutez -le à la *liste des entrées* .
9. Sinon, si l'élément *de champ* est un inputélément dont typel'attribut est à l'état Masqué et que le *nom* est une correspondance ASCII non sensible à la casse pour " charset " :
- Soit *charset* le nom de l'encodage si l'encodage est donné, et " UTF-8" sinon.
  - Créez une entrée avec le *nom* et le *jeu de caractères* , et ajoutez - la à la *liste des entrées* .
10. Sinon, créez une entrée avec le *nom* et la valeur de l'élément *de champ* et ajoutez -la à la *liste d'entrées* .
11. Si l'élément a un dirnameattribut et que la valeur de cet attribut n'est pas la chaîne vide, alors :
- Soit *dirname* la valeur de l' dirnameattribut de l'élément.
  - Soit *dir* la chaîne " ltr" si la directionnalité de l'élément est ' ltr ', et " rtl" sinon (c'est-à-dire lorsque la directionnalité de l'élément est ' rtl ').
  - Créez une entrée avec *dirname* et *dir* , et ajoutez -la à la *liste d'entrées* .

Un élément ne peut avoir un dirname attribut que s'il s'agit d'un textareaélément ou d'un inputélément dont typel'attribut est à l'état Texte ou à l'état Recherche .

6. Soit les données du formulaire un nouvel FormDataobjet associé à la *liste d'entrées* .

7. [Déclenchez un événement](#) nommé [formData](#) au *formulaire* à l'aide de [FormDataEvent](#), avec l' [formData](#) attribut initialisé sur les *données du formulaire* et l' [bubbles](#) attribut initialisé sur *true*.
8. Définissez [la liste d'entrées de construction](#) du *formulaire* sur *false*.
9. Renvoie un [clone](#) de *liste d'entrées* .

#### 4.10.21.5 Sélection d'un encodage de soumission de formulaire

Si l'agent utilisateur doit **choisir un encodage pour un formulaire** , il doit exécuter les étapes suivantes :

1. Soit *encoding* le [codage des caractères du document](#) .
2. Si l' [form](#) élément a un [accept-charset](#) attribut, définissez *encoding* sur la valeur de retour de l'exécution de ces sous-étapes :
  1. Soit *input* la valeur de l' attribut [form](#) de l'élément [accept-charset](#).
  2. Laissez les *étiquettes de codage candidates* être le résultat de [la division de l'entrée sur les espaces blancs ASCII](#) .
  3. Soit les *encodages candidats* une liste vide d' [encodages de caractères](#) .
  4. Pour chaque jeton dans les *étiquettes d'encodage candidates* à tour de rôle (dans l'ordre dans lequel elles ont été trouvées dans *input* ), [obtenez un encodage](#) pour le jeton et, si cela n'entraîne pas d'échec, ajoutez l' [encodage](#) aux *encodages candidats* .
  5. Si les *encodages candidats* sont vides, retournez [UTF-8](#) .
  6. Renvoie le premier encodage dans les *encodages candidats* .
3. Renvoie le résultat de [l'obtention d'un encodage de sortie](#) à partir de *encoding* .

#### 4.10.21.6 Conversion d'une liste d'entrées en une liste de paires nom-valeur

Les algorithmes de codage [application/x-www-form-urlencoded](#) et [text/plain](#) prennent une liste de paires nom-valeur, où les valeurs doivent être des chaînes, plutôt qu'une [liste d'entrées](#) où la valeur peut être un [File](#). L'algorithme suivant effectue la conversion.

Pour **convertir en une liste de paires nom-valeur** une *liste d'entrées* [entry list](#) , exécutez ces étapes :

1. Soit *list* une [liste](#) vide de paires nom-valeur.
2. [Pour chaque](#) *entrée* de *la liste des entrées* :
  1. Soit *name* le [nom](#) de *l'entrée* , avec chaque occurrence de U+000D (CR) non suivie de U+000A (LF), et chaque occurrence de U+000A (LF) non précédée de U+000D (CR), remplacée par une chaîne composée de U+000D (CR) et U+000A (LF).
  2. Si [la valeur](#) de *l'entrée* est un objet, alors laissez *la valeur* être [la valeur](#) de *l'entrée* . Sinon, laissez *value* être [la valeur](#) de *entry* .[Filename](#)
  3. Remplacez chaque occurrence de U+000D (CR) non suivie de U+000A (LF), et chaque occurrence de U+000A (LF) non précédée de U+000D (CR), dans *value* , par une chaîne composée de *U + 000D* (CR) et U+000A (LF).
  4. [Ajouter](#) pour *lister* une nouvelle paire nom-valeur dont le nom est *name* et dont la valeur est *value* .
3. *Liste* de retour .

#### 4.10.21.7 Données de formulaire codées en URL

Voir *l'URL* pour plus de détails sur [application/x-www-form-urlencoded](#). [URL](#)

#### 4.10.21.8 Données de formulaire en plusieurs parties

L' [multipart/form-data](#) **algorithme de codage** , étant donné une *liste d'entrées* [liste d'entrées](#) et un *encodage* [encoding](#) , est le suivant :

1. [Pour chaque](#) *entrée* de *la liste des entrées* :
  1. Remplacez chaque occurrence de U+000D (CR) non suivie de U+000A (LF), et chaque occurrence de U+000A (LF) non précédée de U+000D (CR), dans *nom* de *l'entrée*, [par](#) une *chaîne* composée d'un U+000D (CR) et U+000A (LF).
  2. Si [la valeur](#) de *l'entrée* n'est pas un objet, remplacez chaque occurrence de U+000D (CR) non suivie de U+000A (LF), et chaque

occurrence de U+000A (LF) non précédée de U+000D (CR) , dans [la valeur](#) de l' *entrée* , par une chaîne composée d'un U+000D (CR) et d'un U+000A (LF).[File](#)

2. Renvoie la séquence d'octets résultant du codage de la *liste d'entrées* à l'aide des règles décrites par RFC 7578, *Returning Values from Forms:multipart/form-data* , dans les conditions suivantes : [\[RFC7578\]](#)
  1. Chaque [entrée](#) de la *liste d'entrées* est un *champ* , le [nom](#) de l'entrée est le *nom du champ* et la [valeur](#) de l'entrée est la *valeur du champ* .
  2. L'ordre des pièces doit être le même que l'ordre des champs dans la *liste des entrées* . Plusieurs entrées portant le même nom doivent être traitées comme des champs distincts.
  3. Les noms de champ, les valeurs de champ pour les champs non-fichier et les noms de fichier pour les champs de fichier, dans la [multipart/form-data](#) ressource générée doivent être définis sur le résultat de [l'encodage](#) du nom ou de la valeur de l'entrée correspondante avec *encoding* , convertis en une séquence d'octets.
  4. Pour les noms de champs et les noms de fichiers pour les champs de fichiers, le résultat de l'encodage dans la puce précédente doit être échappé en remplaçant tous les octets 0x0A (LF) par la séquence d'octets ` `, 0x0D (CR) par ` ` et 0x22 (" %0A) %0D par ` %22`. L'agent utilisateur ne doit pas effectuer d'autres échappements.
  5. Les parties de la [multipart/form-data](#) ressource générée qui correspondent à des champs non-fichiers ne doivent pas avoir d' [Content-Type](#) en-tête ` ` spécifié.
  6. La frontière utilisée par l'agent utilisateur pour générer la valeur de retour de cet algorithme est la [multipart/form-data](#) **chaîne de frontière** . (Cette valeur est utilisée pour générer le type MIME de la charge utile de soumission de formulaire générée par cet algorithme.)

Pour plus de détails sur l'interprétation [multipart/form-data](#) des charges utiles, consultez la RFC 7578. [\[RFC7578\]](#)

#### 4.10.21.9 Données de formulaire en texte brut

L' [text/plain](#) **algorithme de codage** , étant donné une liste de paires nom-valeur , est le suivant :

1. Soit *result* la chaîne vide.
2. Pour chaque *paire* par *paires* :



1. Ajouter le nom de *la paire au résultat* .
  2. Ajoutez un seul caractère SIGNE ÉGAL U+003D (=) au *résultat* .
  3. Ajouter la valeur de *la paire au résultat* .
  4. Ajoutez une paire de caractères U+000D RETOUR CHARIOT (CR) U+000A SAUT DE LIGNE (LF) au *résultat* .
3. Retourner *le résultat* .

Les charges utiles utilisant le [text/plain](#) format sont destinées à être lisibles par l'homme. Ils ne sont pas interprétables de manière fiable par ordinateur, car le format est ambigu (par exemple, il n'y a aucun moyen de distinguer une nouvelle ligne littérale dans une valeur de la nouvelle ligne à la fin de la valeur).

#### 4.10.21.10 L' [SubmitEvent](#) interface

✓ MDN  
✓ MDN

[Exposed=Window]

```
interface SubmitEvent : Event {
```

```
  constructor(DOMString type, optional SubmitEventInit
```

```
  eventInitDict = {});
```

```
  readonly attribute HTMLElement? submitter;
```

```
};
```

```
dictionary SubmitEventInit : EventInit {
```

```
  HTMLElement? submitter = null;
```

```
};
```

#### **event.submitter**

Renvoie l'élément représentant le [bouton de soumission](#) qui a déclenché la [soumission du formulaire](#) , ou null si la soumission n'a pas été déclenchée par un bouton.

L' **submitter** attribut doit renvoyer la valeur à laquelle il a été initialisé.

#### 4.10.21.11 L' FormDataEvent interface

✓ MDN

```
[Exposed=Window]
```

```
interface FormDataEvent : Event {
```

```
  constructor(DOMString type, FormDataEventInit eventInitDict);
```

```
  readonly attribute FormData formData;
```

```
};
```

```
dictionary FormDataEventInit : EventInit {
```

```
  required FormData formData;
```

```
};
```

##### event.formData

Renvoie un [FormData](#) objet représentant les noms et les valeurs des éléments associés à la cible [form](#). Les opérations sur l' [FormData](#) objet affecteront les données du formulaire à soumettre.

L' **formData** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente un [FormData](#) objet associé à la [liste d'entrées](#) qui est [construite](#) lorsque le [form](#) est soumis.

#### 4.10.22 Réinitialiser un formulaire

Lorsqu'un *formulaire* [form](#) d'élément est **réinitialisé** , exécutez ces étapes :

1. Soit *reset* le résultat du [déclenchement d'un événement](#) nommé [reset](#) à *form* , avec les attributs [bubbles](#) et [cancelable](#) initialisés à true.

2. Si *reset* est true, appelez l' [algorithme de réinitialisation](#) de chaque [élément réinitialisable](#) dont [le propriétaire du formulaire](#) est *form* .

Chaque [élément réinitialisable](#) définit son propre **algorithme de réinitialisation** . Les modifications apportées aux contrôles de formulaire dans le cadre de ces algorithmes ne comptent pas comme des modifications causées par l'utilisateur (et donc, par exemple, ne provoquent pas le [input](#) déclenchement d'événements).

## 4.11 Éléments interactifs

### 4.11.1 L' [details](#) élément



#### Catégories :

[Contenu du flux](#) .  
[Contenu interactif](#) .  
[Contenu palpable](#) .

#### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu de flux](#) est attendu.

#### Modèle de contenu :

Un [summary](#) élément suivi du [contenu du flux](#) .

#### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

#### Attributs de contenu :

[Attributs globaux](#)  
[open](#) — Si les détails sont visibles

#### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

#### Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLDetailsElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
[CEReactions] attribute boolean open;
```

```
};
```

L' detailsélément représente un widget de divulgation à partir duquel l'utilisateur peut obtenir des informations ou des contrôles supplémentaires.

*L' detailsélément n'est pas approprié pour les notes de bas de page. Veuillez consulter [la section sur les notes de bas de page](#) pour plus de détails sur la façon de marquer les notes de bas de page.*

Le premier summaryélément enfant de l'élément, le cas échéant, représente le résumé ou la légende des détails. S'il n'y a pas summaryd'élément enfant, l'agent utilisateur doit fournir sa propre légende (par exemple "Détails").

Le reste du contenu de l'élément représente les informations ou contrôles supplémentaires.

L' openattribut content est un attribut booléen . S'il est présent, il indique que le résumé et les informations supplémentaires doivent être présentés à l'utilisateur. Si l'attribut est absent, seul le résumé doit être affiché.

Lors de la création de l'élément, si l'attribut est absent, les informations complémentaires doivent être masquées ; si l'attribut est présent, cette information doit être affichée. Par la suite, si l'attribut est supprimé, les informations doivent être masquées ; si l'attribut est ajouté, les informations doivent être affichées.

L'agent utilisateur doit permettre à l'utilisateur de demander que les informations supplémentaires soient affichées ou masquées. Pour honorer une demande d'affichage des détails, l'agent utilisateur doit définir l' openattribut de l'élément sur la chaîne vide. Pour honorer une demande de masquage des informations, l'agent utilisateur doit supprimer l' openattribut de l'élément.

*Cette possibilité de demander que des informations supplémentaires soient affichées ou masquées peut simplement être le comportement d'activation de l' summaryélément approprié, dans le cas où un tel élément existe. Cependant, si aucun élément de ce type n'existe, les agents utilisateurs peuvent toujours fournir cette capacité par le biais d'une autre offre d'interface utilisateur.*

Chaque fois que l' openattribut est ajouté ou supprimé d'un detailsélément, l'agent utilisateur doit mettre en file d'attente une tâche d'élément sur la source de la tâche de manipulation DOM donnée à cet detailsélément qui exécute les étapes suivantes, appelées **étapes de tâche de notification des détails** , pour cet detailsélément :

1. Si une autre tâche a été mise en file d'attente pour exécuter les étapes de la tâche de notification des détails pour cet detailsélément, revenez.

Lorsque l' openattribut est basculé plusieurs fois de suite, ces étapes sont essentiellement fusionnées de sorte qu'un seul événement est déclenché.

2. Lancez un événement nommé toggle à l' detailsélément.

L' open attribut IDL doit refléter l' openattribut content.

L' **algorithme révélant les détails de l'ancêtre** consiste à exécuter les étapes suivantes sur *currentNode* :

1. Alors que *currentNode* a un nœud parent dans l' arborescence plate :
  1. Si *currentNode* est inséré dans le deuxième emplacement d'un details élément :
    1. Définissez *currentNode* sur l' detailsélément dans lequel *currentNode* est inséré.
    2. Si l' openattribut n'est pas défini sur *currentNode* , définissez l' openattribut sur *currentNode* sur la chaîne vide.
  2. Sinon, définissez *currentNode* sur le nœud parent de *currentNode* dans l' arborescence plate .

L'exemple suivant montre l' detailsélément utilisé pour masquer les détails techniques dans un rapport d'avancement.

```
<section class="progress window">
  <h1>Copying "Really Achieving Your Childhood Dreams"</h1>
  <details>
    <summary>Copying... <progress max="375505392"
value="97543282"></progress> 25%</summary>
    <dl>
      <dt>Transfer rate:</dt> <dd>452KB/s</dd>
      <dt>Local filename:</dt> <dd>/home/rpausch/raycd.m4v</dd>
      <dt>Remote filename:</dt> <dd>/var/www/lectures/raycd.m4v</dd>
      <dt>Duration:</dt> <dd>01:16:27</dd>
      <dt>Color profile:</dt> <dd>SD (6-1-6)</dd>
      <dt>Dimensions:</dt> <dd>320×240</dd>
    </dl>
  </details>
</section>
```

Ce qui suit montre comment un detailsélément peut être utilisé pour masquer certains contrôles par défaut :

```
<details>
```

```
<summary><label for=fn>Name & Extension:</label></summary>
<p><input type=text id=fn name=fn value="Pillar Magazine.pdf">
<p><label><input type=checkbox name=ext checked> Hide
extension</label>
</details>
```

On pourrait l'utiliser en conjonction avec d'autres details dans une liste pour permettre à l'utilisateur de réduire un ensemble de champs à un petit ensemble d'entêtes, avec la possibilité d'ouvrir chacun d'eux.





Dans ces exemples, le résumé résume vraiment ce que les contrôles peuvent changer, et non les valeurs réelles, ce qui est loin d'être idéal.

Étant donné que l' `open` attribut est ajouté et supprimé automatiquement lorsque l'utilisateur interagit avec le contrôle, il peut être utilisé dans CSS pour styliser l'élément différemment en fonction de son état. Ici, une feuille de style permet d'animer la couleur du résumé à l'ouverture ou à la fermeture de l'élément :

```
<style>
  details > summary { transition: color 1s; color: black; }
  details[open] > summary { color: red; }
</style>
```



```
<details>
  <summary>Automated Status: Operational</summary>
  <p>Velocity: 12m/s</p>
  <p>Direction: North</p>
</details>
```

#### 4.11.2 L' **summary** élément



##### Catégories :

Aucun.

##### Contextes dans lesquels cet élément peut être utilisé :

En tant que premier enfant d'un details élément.

##### Modèle de contenu :

Contenu de la phrase , éventuellement mélangé avec le contenu du titre .

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

Attributs globaux

##### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

##### Interface DOM :

Utilisations HTMLElement.

L' **summary** élément représente un résumé, une légende ou une légende pour le reste du contenu de l' élément **summary** parent de l'élément details , le cas échéant.

Un **summary** élément est un **résumé des détails de son parent** si l'algorithme suivant renvoie true :

1. Si cet **summary** élément n'a pas de parent, alors retourne false.
2. Soit *parent* le parent de cet **summary** élément.
3. Si *parent* n'est pas un details élément, renvoie false.
4. Si le premier élément enfant du *parent* **summary** n'est pas cet **summary** élément, alors renvoie false.

5. Renvoie vrai.

Le [comportement d'activation](#) des [summary](#) éléments consiste à exécuter les étapes suivantes :

1. Si cet [summary](#) élément n'est pas le [résumé de ses détails parents](#) , alors retournez.
2. Soit *parent* le parent de cet [summary](#) élément.
3. Si l' [open](#) attribut est présent sur *parent* , [supprimez](#) -le. Sinon, [définissez](#) l' attribut [parentopen](#) sur la chaîne vide.

*Cela exécutera ensuite les [étapes de la tâche de notification des détails](#) .*

### 4.11.3 Commandes

#### 4.11.3.1 Facettes

Une **commande** est l'abstraction derrière les éléments de menu, les boutons et les liens. Une fois qu'une commande est définie, d'autres parties de l'interface peuvent faire référence à la même commande, permettant à de nombreux points d'accès à une même fonctionnalité de partager des facettes telles que l' [état désactivé](#) .

Les commandes sont définies pour avoir les **facettes** suivantes :

##### ***Étiqueter***

Le nom de la commande tel qu'il est vu par l'utilisateur.

##### ***Clef d'accès***

Une combinaison de touches sélectionnée par l'agent utilisateur qui déclenche la commande. Une commande peut ne pas avoir de clé d'accès.

##### ***État caché***

Si la commande est masquée ou non (essentiellement, si elle doit être affichée dans les menus).

##### ***État désactivé***

Si la commande est pertinente et peut être déclenchée ou non.

##### ***Action***

L'effet réel qu'aura le déclenchement de la commande. Il peut s'agir d'un gestionnaire d'événements scripté, d'une [URL](#) vers laquelle [naviguer](#) ou d'une soumission de formulaire.

Les agents utilisateurs peuvent exposer les [commandes](#) qui correspondent aux critères suivants :

- La facette État caché est fausse (visible)
- L'élément se trouve dans un document avec un contexte de navigation non nul .
- Ni l'élément ni aucun de ses ancêtres n'a d' hidden attribut spécifié.

Les agents utilisateurs sont encouragés à le faire, en particulier pour les commandes qui ont des clés d'accès , afin d'annoncer ces clés à l'utilisateur.

Par exemple, de telles commandes pourraient être répertoriées dans la barre de menus de l'agent utilisateur.

#### 4.11.3.2 Utilisation de l' **a**élément pour définir une commande

Un aélément avec un hrefattribut définit une commande .

Le Label de la commande est le contenu textuel descendant de l'élément .

La clé d'accès de la commande est la clé d'accès attribuée à l'élément , le cas échéant.

L' état caché de la commande est vrai (caché) si l'élément a un hiddenattribut, et faux sinon.

La facette Disabled State de la commande est true si l'élément ou l'un de ses ancêtres est inert , et false sinon.

L' action de la commande est de déclencher un clickévénement sur l'élément.

#### 4.11.3.3 Utilisation de l' **button**élément pour définir une commande

Un buttonélément définit toujours une commande .

Les facettes Label , Access Key , Hidden State et Action de la commande sont déterminées comme pour a les éléments (voir la section précédente).

L' état désactivé de la commande est true si l'élément ou l'un de ses ancêtres est inert , ou si l' état désactivé de l'élément est défini, et false sinon.

#### 4.11.3.4 Utilisation de l' **input**élément pour définir une commande

Un input élément dont type l'attribut est dans l'un des états Bouton Soumettre , Bouton Réinitialiser , Bouton Image , Bouton , Bouton Radio ou Case à cocher définit une commande .

Le libellé de la commande est déterminé comme suit :

- Si l' type attribut est dans l'un des états Submit Button , Reset Button , Image Button ou Button , alors le Label est la chaîne donnée par l' value attribut, le cas échéant, et une valeur dépendante de l'UA et des paramètres régionaux que l'UA utilise pour étiqueter le bouton lui-même si l'attribut est absent.
- Sinon, si l'élément est un contrôle étiqueté , alors Label est le contenu textuel descendant du premier label élément dans l'arborescence dont le contrôle étiqueté est l'élément en question. (En termes JavaScript, ceci est donné par `element.labels[0].textContent.`)
- Sinon, si l' value attribut est présent, alors le Label est la valeur de cet attribut.
- Sinon, le Label est la chaîne vide.

*Même si l' value attribut sur input les éléments dans l' état du bouton d'image n'est pas conforme, l'attribut peut toujours contribuer à la détermination de l'étiquette , s'il est présent et que l' alt attribut du bouton d'image est manquant.*

La clé d'accès de la commande est la clé d'accès attribuée à l'élément , le cas échéant.

L' état caché de la commande est vrai (caché) si l'élément a un hidden attribut, et faux sinon.

L' état désactivé de la commande est true si l'élément ou l'un de ses ancêtres est inert , ou si l' état désactivé de l'élément est défini, et false sinon.

L' action de la commande est de déclencher un click événement sur l'élément.

#### 4.11.3.5 Utilisation de l' option élément pour définir une commande

Un option élément avec un select élément ancêtre et soit aucun value attribut, soit un value attribut qui n'est pas la chaîne vide définit une commande .

Le Label de la commande est la valeur de l' attribut option de l'élément , s'il y en a un, ou bien le contenu textuel descendant de l'élément , avec les espaces ASCII supprimés et réduits .labeloption

La [clé d'accès](#) de la commande est la [clé d'accès attribuée](#) à l'élément , le cas échéant.

L' [état caché](#) de la commande est vrai (caché) si l'élément a un [hidden](#)attribut, et faux sinon.

L' [état Disabled](#) de la commande est true si l'élément est [disabled](#) , ou si son [select](#)élément ancêtre le plus proche est [disabled](#) , ou si lui ou l'un de ses ancêtres est [inert](#) , et false sinon.

Si l' élément [option](#)ancêtre le plus proche de a un attribut, l' [action](#) de la commande est de [basculer](#) l' élément. Sinon, l' [action](#) consiste à [sélectionner](#) l' élément.[selectmultipleoptionoption](#)

#### 4.11.3.6 Utilisation de l' [accesskey](#)attribut sur un [legend](#)élément pour définir une commande

Un [legend](#)élément [définit une commande](#) si toutes les conditions suivantes sont vraies :

- Il a une [clé d'accès assignée](#) .
- C'est un enfant d'un [fieldset](#)élément.
- Son parent a un descendant qui [définit une commande](#) qui n'est ni un [label](#)élément ni un [legend](#)élément. Cet élément, s'il existe, est le **délégataire [legend](#)de l'élément[accesskey](#)** .

Le [Label](#) de la commande est le [contenu textuel descendant](#) de l'élément .

La [clé d'accès](#) de la commande est la [clé d'accès attribuée](#) à l'élément .

Les facettes [État masqué](#) , [État désactivé](#) et [Action](#) de la commande sont les mêmes que les facettes respectives du [délégué de l' \[legend\]\(#\)élément\[accesskey\]\(#\)](#) .

Dans cet exemple, l' [legend](#)élément spécifie un [accesskey](#), qui, lorsqu'il est activé, déléguera à l' [input](#)élément à l'intérieur de l' [legend](#)élément.

```
<fieldset>
  <legend accesskey=p>
    <label>I want <input name=pizza type=number step=1 value=1 min=0>
      pizza(s) with these toppings</label>
  </legend>
  <label><input name=pizza-cheese type=checkbox checked>
    Cheese</label>
```

```
<label><input name=pizza-ham type=checkbox checked> Ham</label>
<label><input name=pizza-pineapple type=checkbox>
Pineapple</label>
</fieldset>
```

#### 4.11.3.7 Utilisation de l' **accesskey** attribut pour définir une commande sur d'autres éléments

Un élément auquel une [clé d'accès est attribuée définit une commande](#) .

Si l'une des sections précédentes qui définissent des éléments qui [définissent des commandes](#) définissent que cet élément [définit une commande](#) , alors cette section s'applique à cet élément, et cette section ne s'applique pas. Sinon, cette section s'applique à cet élément.

Le [Libellé](#) de la commande dépend de l'élément. Si l'élément est un [contrôle étiqueté](#) , le [contenu textuel descendant](#) du premier [label](#) élément dans [l'arborescence](#) dont [le contrôle étiqueté](#) est l'élément en question est le [Label](#) (en termes JavaScript, il est donné par `element.labels[0].textContent`). Sinon, [Label](#) est le [contenu textuel descendant](#) de l'élément .

La [clé d'accès](#) de la commande est la [clé d'accès attribuée](#) à l'élément .

L' [état caché](#) de la commande est vrai (caché) si l'élément a un [hidden](#) attribut, et faux sinon.

L' [état désactivé](#) de la commande est true si l'élément ou l'un de ses ancêtres est [inert](#) , et false sinon.

L' [action](#) de la commande consiste à exécuter les étapes suivantes :

1. Exécutez les [étapes de mise au point](#) pour l'élément.
2. [Déclenche un click](#) événement sur l'élément.

#### 4.11.4 L' **dialog** élément



**Catégories :**

[Contenu du flux](#) .

## Contextes dans lesquels cet élément peut être utilisé :

Où le contenu de flux est attendu.

## Modèle de contenu :

Contenu du flux .

## Omission de balise dans text/html :

Aucune des deux balises n'est omise.

## Attributs de contenu :

Attributs globaux

open — Si la boîte de dialogue s'affiche

## Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

## Interface DOM :

```
[Exposed=Window]
```

```
interface HTMLDialogElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute boolean open;
```

```
  attribute DOMString returnValue;
```

```
  [CEReactions] undefined show();
```

```
  [CEReactions] undefined showModal();
```

```
  [CEReactions] undefined close(optional DOMString
```

```
  returnValue);
```

```
};
```

L' dialog élément représente une partie transitoire d'une application, sous la forme d'une petite fenêtre ("boîte de dialogue"), avec laquelle l'utilisateur interagit pour effectuer une tâche ou recueillir des informations. Une fois que l'utilisateur a terminé, la boîte de dialogue peut être fermée automatiquement par l'application ou fermée manuellement par l'utilisateur.

En particulier pour les boîtes de dialogue modales, qui sont un modèle familier dans tous les types d'applications, les auteurs doivent veiller à ce que les boîtes de

dialogue de leurs applications Web se comportent d'une manière familière aux utilisateurs d'applications non Web.

*Comme avec tous les éléments HTML, il n'est pas conforme d'utiliser l'`dialog` élément lorsque vous essayez de représenter un autre type de contrôle. Par exemple, les menus contextuels, les info-bulles et les listes déroulantes contextuelles ne sont pas des boîtes de dialogue, `dialog` il est donc incorrect d'abuser de l'élément pour implémenter ces modèles.*

Une partie importante du comportement du dialogue face à l'utilisateur est le placement du focus initial. Les [étapes de mise au point de la boîte de dialogue](#) tentent de choisir un bon candidat pour la mise au point initiale lorsqu'une boîte de dialogue est affichée, mais peuvent ne pas remplacer les auteurs qui réfléchissent soigneusement au choix correct pour correspondre aux attentes de l'utilisateur pour une boîte de dialogue spécifique. Ainsi, les auteurs doivent utiliser l'`autofocus` attribut sur l'élément descendant de la boîte de dialogue avec lequel l'utilisateur est censé interagir immédiatement après l'ouverture de la boîte de dialogue. S'il n'y a pas un tel élément, les auteurs doivent utiliser l'`autofocus` attribut sur l'`dialog` élément lui-même.

Dans l'exemple suivant, une boîte de dialogue est utilisée pour modifier les détails d'un produit dans une application Web de gestion des stocks.

```
<dialog>
  <label>Product Number <input type="text" readonly></label>
  <label>Product Name <input type="text" autofocus></label>
</dialog>
```

Si l'`autofocus` attribut n'était pas présent, le champ Numéro de produit aurait été ciblé par les étapes de focalisation de la boîte de dialogue. Bien qu'il s'agisse d'un comportement raisonnable, l'auteur a déterminé que le champ le plus pertinent à cibler était le champ Nom du produit, car le champ Numéro du produit est en lecture seule et n'attend aucune entrée de l'utilisateur. Ainsi, l'auteur a utilisé la mise au point automatique pour remplacer la valeur par défaut.

Même si l'auteur souhaite mettre l'accent sur le champ Numéro de produit par défaut, il est préférable de le spécifier explicitement en utilisant la mise au point automatique sur cet `input` élément. Cela rend l'intention évidente pour les futurs lecteurs du code et garantit que le code reste robuste face aux futures mises à jour. (Par exemple, si un autre développeur a ajouté un bouton de fermeture et l'a positionné dans l'arborescence des nœuds avant le champ Numéro de produit).

Un autre aspect important du comportement de l'utilisateur est de savoir si les boîtes de dialogue peuvent défiler ou non. Dans certains cas, le débordement (et donc la capacité de défilement) ne peut être évité, par exemple, lorsqu'il est causé par les paramètres de zoom de texte élevés de l'utilisateur. Mais en général, les utilisateurs ne s'attendent pas à des boîtes de dialogue déroulantes. L'ajout de nœuds de texte volumineux directement aux éléments de dialogue est particulièrement mauvais car



cela risque de provoquer un débordement de l'élément de dialogue lui-même. Les auteurs feraient mieux de les éviter.

La boîte de dialogue des conditions d'utilisation suivante respecte les suggestions ci-dessus.

```
<dialog style="height: 80vh;">
  <div style="overflow: auto; height: 60vh;" autofocus>
    <p>By placing an order via this Web site on the first day of
    the fourth month of the year
      2010 Anno Domini, you agree to grant Us a non-transferable
    option to claim, for now and for
      ever more, your immortal soul.</p>
    <p>Should We wish to exercise this option, you agree to
    surrender your immortal soul,
      and any claim you may have on it, within 5 (five) working days
    of receiving written
      notification from this site or one of its duly authorized
    minions.</p>
    <!-- ... etc., with many more <p> elements ... -->
  </div>
  <form method="dialog">
    <button type="submit" value="agree">Agree</button>
    <button type="submit" value="disagree">Disagree</button>
  </form>
</dialog>
```

Notez comment les [étapes de mise au point de la boîte de dialogue](#) auraient sélectionné l' [div](#)élément défilant par défaut, mais comme dans l'exemple précédent, nous avons placé [autofocus](#) le [div](#) afin d'être plus explicite et robuste face aux modifications futures.

En revanche, si les [p](#)éléments exprimant les conditions d'utilisation n'avaient pas un tel [div](#)élément wrapper, alors le [dialog](#) lui-même deviendrait défilable, enfreignant les conseils ci-dessus. De plus, en l'absence de tout [autofocus](#) attribut, un tel modèle de balisage aurait enfreint les conseils ci-dessus et déclenché le comportement par défaut [des étapes de mise au point de la boîte de dialogue](#), et aurait fait passer le focus à Agree [button](#), ce qui est une mauvaise expérience utilisateur.

L' [open](#) attribut est un [attribut booléen](#). Lorsqu'il est spécifié, il indique que l' [dialog](#) élément est actif et que l'utilisateur peut interagir avec lui.

Un [dialog](#) élément sans [open](#) attribut spécifié ne doit pas être montré à l'utilisateur. Cette exigence peut être mise en œuvre indirectement via la couche de style. Par exemple, les agents utilisateurs qui [prennent en charge le rendu par défaut](#)

[suggéré](#) implémentent cette exigence en utilisant les règles CSS décrites dans la [section Rendu](#) .

*La suppression de l' [open](#) attribut masque généralement la boîte de dialogue. Cependant, cela a un certain nombre de conséquences supplémentaires étranges :*

- L' [close](#) événement ne sera pas déclenché.
- La [close\(\)](#) méthode et toute [interface d'annulation fournie par l'agent utilisateur](#) ne pourront plus fermer la boîte de dialogue.
- Si la boîte de dialogue a été affichée à l'aide de sa [showModal\(\)](#) méthode, le [Document](#) sera toujours [bloqué](#) .

*Pour ces raisons, il est généralement préférable de ne jamais supprimer l' [open](#) attribut manuellement. Utilisez plutôt la [close\(\)](#) méthode pour fermer la boîte de dialogue ou l' [hidden](#) attribut pour la masquer.*

L' [tabindex](#) attribut ne doit pas être spécifié sur [dialog](#) les éléments.

**`dialog.show()`**

✓ 

Affiche l' [dialog](#) élément.

**`dialog.showModal()`**

✓ 

Affiche l' [dialog](#) élément et en fait la boîte de dialogue modale la plus élevée.  
Cette méthode respecte l' [autofocus](#) attribut.

**`dialog.close([ result ])`**

✓ 

Ferme l' [dialog](#) élément.  
L'argument, s'il est fourni, fournit une valeur de retour.

**`dialog.returnValue [ = result ]`**

✓ 

Renvoie la [dialog](#) valeur de retour de .  
Peut être défini pour mettre à jour la valeur de retour.

Lorsque la [show\(\)](#) méthode est invoquée, l'agent utilisateur doit exécuter les étapes suivantes :

1. Si l'élément a déjà un [open](#) attribut, alors retournez.

2. Si *le sujet* est dans l' [état d'affichage du popover](#) , lancez un `"InvalidStateError"` [DOMException](#) .
3. Ajoutez un `open` attribut à l' `dialog` élément, dont la valeur est la chaîne vide.
4. Définissez l' [élément précédemment focalisé](#) `dialog` de l'élément sur l'élément [focalisé](#) .
5. Exécutez les [étapes de focalisation de la boîte de dialogue](#) en fonction de l' `dialog` élément et de la valeur false.

Lorsque la `showModal()` méthode est invoquée, l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit *sujet* l' `dialog` élément sur lequel la méthode a été invoquée.
2. Si *le sujet* a déjà un `open` attribut, lancez un `"InvalidStateError"` [DOMException](#) .
3. Si *le sujet* n'est pas [connecté](#) , lancez un `"InvalidStateError"` [DOMException](#) .
4. Si *le sujet* est dans l' [état d'affichage du popover](#) , lancez un `"InvalidStateError"` [DOMException](#) .
5. Ajoutez un `open` attribut à `subject` , dont la valeur est la chaîne vide.
6. Définissez l' indicateur [is modal of](#) `subject` sur true.
7. Laissez [le document du nœud](#) du *sujet* être [bloqué par le](#) *sujet* du dialogue modal .

*Cela rendra la [zone focalisée du document](#) inerte (à moins que cette zone actuellement focalisée ne soit un [descendant](#) de *sujet* [incluant une ombre](#) ). Dans de tels cas, la [zone focalisée du document](#) sera bientôt [réinitialisée](#) à la [fenêtre](#) . En quelques étapes, nous tenterons de trouver un meilleur candidat sur lequel nous concentrer.*

8. Si [la couche supérieure](#) du [document du nœud](#) du *sujet* ne [contient pas déjà](#) *le sujet* , alors [ajoutez](#) *le sujet* à [la couche supérieure](#) du document du [nœud du](#) *sujet* .
9. Définissez l' [élément précédemment focalisé](#) du *sujet* sur l'élément [focalisé](#) .
10. Exécutez les [étapes de mise au point de la boîte de dialogue](#) en fonction du *sujet* et de la valeur true.

Les **étapes de mise au point de la boîte de dialogue** , étant donné un `dialog` élément `subject` et un booléen `isModal` , sont les suivantes :

1. Exécutez masquer tous les popovers jusqu'à ce que null, false et false soient donnés.
2. Soit *le contrôle* nul.
3. Si *isModal* est true et que *subject* a l' autofocus attribut, alors définissez *control* sur *subject* .
4. Si *control* est null, définissez *control* sur le délégué de focus de *subject* .
5. Si *control* est null, définissez *control* sur *subject* .
6. Exécutez les étapes de mise au point pour *le contrôle* .

*Si le contrôle n'est pas focalisable , cela ne fera rien. Cela ne se produirait que si le sujet n'avait pas de délégué de focus et que l'agent utilisateur décidait que dialog les éléments n'étaient généralement pas focalisables. Dans ce cas, toutes les modifications antérieures apportées à la zone ciblée du document s'appliqueront.*

7. Soit *topDocument* le document actif du nœud navigable du contrôle de niveau supérieur traversable .
8. Si l'origine du document du nœud du contrôle n'est pas la même que l' origine de *topDocument* , alors retournez.
9. Vide les candidats autofocus de *topDocument* .
10. Définissez l'indicateur de traitement de mise au point automatique de *topDocument* sur true.

Si à tout moment un dialog élément est supprimé d'un Document , alors s'il se dialog trouve dans cette Document couche supérieure , il doit en être supprimé . Définissez également l' indicateur is modal dialog de l'élément sur false.

Lorsque la méthode est invoquée, l'agent utilisateur doit fermer la boîte de dialogue sur laquelle la méthode a été invoquée. Si *returnValue* a été donné, il doit être utilisé comme valeur de retour ; sinon, il n'y a pas de valeur de retour. **close (returnValue)**

Lorsqu'un dialog élément *subject* doit être **fermé** , éventuellement avec une valeur de retour *result* , l'agent utilisateur doit exécuter les étapes suivantes :

1. Si *le sujet* n'a pas d' open attribut, alors retournez.
2. Supprimer l'attribut du *sujet* open .
3. Définissez l' indicateur is modal of *subject* sur false.
4. Si l'argument *result* a été fourni, définissez l' returnValue attribut sur la valeur de *result* .

5. Si le *sujet* est dans son Document calque supérieur , supprimez -le.
6. Si l'élément précédemment focalisé du *sujet* n'est pas nul, alors :
  1. Soit *element* l' élément précédemment focalisé du *sujet* .
  2. Définissez l'élément précédemment focalisé du *sujet* sur null.
  3. Exécutez les étapes de mise au point pour *l'élément* ; la fenêtre ne doit pas défiler en effectuant cette étape.
7. Mettez en file d'attente une tâche d'élément sur la source de tâche d'interaction utilisateur en fonction de l' élément *sujet* pour déclencher un événement nommé closesujet .

L' **returnValue**attribut IDL, lors de l'obtention, doit renvoyer la dernière valeur à laquelle il a été défini. Lors du réglage, il doit être réglé sur la nouvelle valeur. Lorsque l'élément est créé, il doit être défini sur la chaîne vide.

*Nous utilisons afficher/fermer comme verbes pour dialog les éléments, par opposition aux paires de verbes qui sont plus communément considérées comme des antonymes tels que afficher/masquer ou ouvrir/fermer, en raison des contraintes suivantes :*

- Masquer une boîte de dialogue est différent de la fermer. La fermeture d'une boîte de dialogue lui donne une valeur de retour, déclenche un événement, débloque la page pour d'autres boîtes de dialogue, etc. Alors que masquer une boîte de dialogue est une propriété purement visuelle, et c'est quelque chose que vous pouvez déjà faire avec l' hiddenattribut ou en supprimant l' openattribut. (Voir également la note ci-dessus sur la suppression de l' openattribut et sur le fait qu'il n'est généralement pas souhaitable de masquer la boîte de dialogue de cette manière.)*
- Afficher une boîte de dialogue est différent de l'ouvrir. L'ouverture d'une boîte de dialogue consiste à créer et à afficher cette boîte de dialogue (similaire à window.open() la création et à l'affichage d'une nouvelle fenêtre). Alors que l'affichage de la boîte de dialogue consiste à prendre un dialogélément qui se trouve déjà dans le DOM et à le rendre interactif et visible pour l'utilisateur.*
- Si nous devons avoir une dialog.open() méthode malgré ce qui précède, cela entrerait en conflit avec la dialog.openpropriété.*

*En outre, une enquête sur de nombreux autres cadres d'interface utilisateur contemporains de la conception originale de l' dialog élément a clairement montré que la paire de verbes afficher/fermer était raisonnablement courante.*

*En résumé, il s'avère que les implications de certains verbes, et la façon dont ils sont utilisés dans des contextes technologiques, signifient que des actions jumelées telles*

que montrer et fermer une boîte de dialogue ne sont pas toujours exprimables comme des antonymes.

---

**Annulation de boîtes de dialogue** : lorsque Document est bloqué par une boîte de dialogue modale, les agents utilisateurs peuvent fournir une interface utilisateur qui, lors de l'activation, met en file d'attente une tâche d'élément sur la source de la tâche d'interaction utilisateur en fonction de l'élément de boîte de dialogue pour exécuter ces étapes :

1. Soit *close* le résultat du déclenchement d'un événement nommé cancel à *dialog*, avec l'cancelable attribut initialisé à *true*.
2. Si *close* est *true* et que *dialog* a un open attribut, alors fermez la boîte de dialogue sans valeur de retour.

Un exemple d'un tel mécanisme d'interface utilisateur serait l'utilisateur appuyant sur la touche "Échap".

---

Chaque dialog élément a un indicateur **is modal**. Lorsqu'un dialog élément est créé, ce drapeau doit être défini sur faux.

Chaque élément HTML a un **élément précédemment focalisé** qui est nul ou un élément, et il est initialement nul. Lorsque showModal() et show() sont appelés, cet élément est défini sur l'élément actuellement ciblé avant d'exécuter les étapes de focalisation de la boîte de dialogue. Les éléments avec l'popover attribut définissent cet élément sur l'élément actuellement ciblé pendant l'algorithme d'affichage du popover.

---



L' open attribut IDL doit refléter l' open attribut content.

Cette boîte de dialogue comporte quelques petits caractères. L' strong élément est utilisé pour attirer l'attention de l'utilisateur sur la partie la plus importante.

```
<dialog>
  <h1>Add to Wallet</h1>
  <p><strong><label for=amt>How many gold coins do you want to add
to your wallet?</label></strong></p>
  <p><input id=amt name=amt type=number min=0 step=0.01
value=100></p>
  <p><small>You add coins at your own risk.</small></p>
  <p><label><input name=round type=checkbox> Only add perfectly
round coins</label></p>
  <p><input type=button onclick="submit()" value="Add Coins"></p>
</dialog>
```

## 4.12 Script

Les scripts permettent aux auteurs d'ajouter de l'interactivité à leurs documents.

Les auteurs sont encouragés à utiliser des alternatives déclaratives aux scripts dans la mesure du possible, car les mécanismes déclaratifs sont souvent plus maintenables et de nombreux utilisateurs désactivent les scripts.

Par exemple, au lieu d'utiliser un script pour afficher ou masquer une section afin d'afficher plus de détails, l' details élément peut être utilisé.

Les auteurs sont également encouragés à faire en sorte que leurs applications se dégradent gracieusement en l'absence de support de script.

Par exemple, si un auteur fournit un lien dans un en-tête de tableau pour reclasser dynamiquement le tableau, le lien peut également fonctionner sans scripts en demandant le tableau trié au serveur.

### 4.12.1 L' script élément



#### Catégories :

[Contenu des métadonnées](#) .

[Contenu du flux](#) .

[Contenu de la phrase](#) .

Élément de support de script .

### Contextes dans lesquels cet élément peut être utilisé :

- Où le contenu des métadonnées est attendu.
- Où le contenu du phrasé est attendu.
- Où les éléments de support de script sont attendus.

### Modèle de contenu :

S'il n'y a pas src d'attribut, cela dépend de la valeur de l' type attribut, mais doit correspondre aux restrictions de contenu du script .  
S'il existe un src attribut, l'élément doit être vide ou contenir uniquement une documentation de script qui correspond également aux restrictions de contenu de script .

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

#### Attributs globaux

src— Adresse de la ressource

type— Type de scénario

nomodule— Empêche l'exécution dans les agents utilisateurs qui prennent en charge les scripts de module

async- Exécuter le script lorsqu'il est disponible, sans bloquer lors de la récupération

defer— Différer l'exécution du script

crossorigin— Comment l'élément gère les requêtes crossorigin

integrity— Métadonnées d'intégrité utilisées dans les contrôles *d'intégrité des sous-ressources* [SRI]

referrerpolicy— Politique de référence pour les récupérations initiées par l'élément

blocking— Si l'élément est potentiellement bloquant le rendu

### Considérations d'accessibilité :

Pour les auteurs .

Pour les exécutants .

### Interface DOM :

[Exposed=Window]

```
interface HTMLScriptElement : HTMLElement {
```

```
    [HTMLConstructor] constructor();
```

```
    [CEReactions] attribute USVString src;
```

```
    [CEReactions] attribute DOMString type;
```



```

[CEReactions] attribute boolean noModule;

[CEReactions] attribute boolean async;

[CEReactions] attribute boolean defer;

[CEReactions] attribute DOMString? crossOrigin;

[CEReactions] attribute DOMString text;

[CEReactions] attribute DOMString integrity;

[CEReactions] attribute DOMString referrerPolicy;

[SameObject, PutForwards=value] readonly attribute
DOMTokenList blocking;

static boolean supports(DOMString type);

// also has obsolete members

};

```

L' **script** élément permet aux auteurs d'inclure un script dynamique et des blocs de données dans leurs documents. L'élément ne représente pas le contenu pour l'utilisateur.



L' **type** attribut permet de personnaliser le type de script représenté :

- Omettre l'attribut, le définir sur la chaîne vide ou le définir sur une correspondance d'essence de type JavaScript MIME , signifie que le script est un script classique , à interpréter conformément à la production de niveau supérieur du script JavaScript. Les scripts classiques sont affectés par les attributs async et defer , mais uniquement lorsque l' src attribut est défini. Les auteurs doivent omettre l' type attribut au lieu de le définir de manière redondante.
- La définition de l'attribut sur une correspondance ASCII non sensible à la casse pour " `module` " signifie que le script est un script de module JavaScript , à interpréter en fonction de la production de niveau supérieur du module

JavaScript. Les scripts de module ne sont pas affectés par l' defer attribut, mais sont affectés par l' async attribut (quel que soit l'état de l' src attribut).

- Définir l' attribut sur une correspondance ASCII non sensible à la casse`importmap` pour " " signifie que le script est une carte d' importation , contenant du JSON qui sera utilisé pour contrôler le comportement de la résolution du spécificateur de module . Les cartes d'importation ne peuvent être qu'en ligne, c'est-à-dire que l' src attribut et la plupart des autres attributs n'ont pas de sens et ne doivent pas être utilisés avec eux.
- Définir l'attribut sur toute autre valeur signifie que le script est un **bloc de données** , qui n'est pas traité. Aucun des script attributs (sauf type lui-même) n'a d'effet sur les blocs de données. Les auteurs doivent utiliser une chaîne de type MIME valide qui n'est pas une correspondance d'essence de type MIME JavaScript pour désigner les blocs de données.

*L'exigence selon laquelle les blocs de données doivent être indiqués à l'aide d'une chaîne de type MIME valide est en place pour éviter d'éventuelles collisions futures. Si jamais cette spécification ajoute des types de script supplémentaires , ils seront déclenchés en définissant l' type attribut sur quelque chose qui n'est pas un type MIME, comme la façon dont la `module` valeur " " indique les scripts de module . En utilisant maintenant une chaîne de type MIME valide, vous vous assurez que votre bloc de données ne sera jamais réinterprété comme un type de script différent, même dans les futurs agents utilisateurs.*

Les scripts classiques et les scripts de module JavaScript peuvent être intégrés en ligne ou importés à partir d'un fichier externe à l'aide de l' src attribut qui, s'il est spécifié, donne l' URL de la ressource de script externe à utiliser. Si src est spécifié, il doit s'agir d'une URL non vide valide potentiellement entourée d'espaces .

Le contenu des script éléments en ligne, ou la ressource de script externe, doit être conforme aux exigences des productions de script ou de module de la spécification JavaScript, respectivement pour les scripts classiques et les scripts de module JavaScript . [JAVASCRIPT]

Le contenu de la ressource de script externe pour les scripts de module CSS doit être conforme aux exigences de la spécification CSS. [CSS]

Le contenu de la ressource de script externe pour les scripts de module JSON doit être conforme aux exigences de la spécification JSON [JSON] .

Le contenu des script éléments en ligne pour les cartes d'importation doit être conforme aux exigences de création de carte d'importation .

Pour les éléments cartographiques importés script , les attributs src, async, nomodule, defer, crossorigin, integrity et ne doivent pas être spécifiés. referrerpolicy

Un document ne doit pas contenir plus d'un élément [cartographique d'importation script](#).

Lorsqu'elles sont utilisées pour inclure [des blocs de données](#), les données doivent être intégrées en ligne, le format des données doit être indiqué à l'aide de l' [type](#) attribut et le contenu de l' [script](#) élément doit être conforme aux exigences définies pour le format utilisé. Les attributs [src](#), [async](#), [nomodule](#), , , et ne doivent pas être [defer](#) spécifiés. [crossoriginintegrityreferrerpolicy](#)

L' [nomodule](#) attribut est un [attribut booléen](#) qui empêche l'exécution d'un script dans les agents utilisateurs prenant en charge [les scripts de module](#). Cela permet une exécution sélective des [scripts de module](#) dans les agents utilisateurs modernes et [des scripts classiques](#) dans les agents utilisateurs plus anciens, [comme indiqué ci-dessous](#). L' [nomodule](#) attribut ne doit pas être spécifié sur [les scripts de module](#) (et sera ignoré s'il l'est).



Les attributs [async](#) et [defer](#) sont [des attributs booléens](#) qui indiquent comment le script doit être évalué. [Les scripts classiques](#) peuvent spécifier [defer](#) ou [async](#), mais ne doivent pas spécifier l'un ou l'autre à moins que l' [src](#) attribut ne soit présent. [Les scripts de module](#) peuvent spécifier l' [async](#) attribut, mais ne doivent pas spécifier l' [defer](#) attribut.

Il existe plusieurs modes possibles qui peuvent être sélectionnés à l'aide de ces attributs, et selon le type de script.

Pour [les scripts classiques](#), si l' [async](#) attribut est présent, le script classique sera récupéré [parallèlement](#) à l'analyse et évalué dès qu'il sera disponible (potentiellement avant la fin de l'analyse). Si l' [async](#) attribut n'est pas présent mais que l' [defer](#) attribut est présent, alors le script classique sera récupéré [en parallèle](#) et évalué une fois l'analyse de la page terminée. Si aucun attribut n'est présent, le script est extrait et évalué immédiatement, bloquant l'analyse jusqu'à ce qu'ils soient tous les deux terminés.

Pour [les scripts de module](#), si l' [async](#) attribut est présent, le script de module et toutes ses dépendances seront récupérés [en parallèle](#) à l'analyse, et le script de module sera évalué dès qu'il sera disponible (potentiellement avant la fin de l'analyse). Sinon, le script du module et ses dépendances seront récupérés [parallèlement](#) à l'analyse et évalués lorsque la page aura terminé l'analyse. (L' [defer](#) attribut n'a aucun effet sur les scripts de module.)

Tout est résumé dans le schéma de principe suivant :

Les détails de traitement exacts de ces attributs sont, pour la plupart des raisons historiques, quelque peu non triviaux, impliquant un certain nombre d'aspects du HTML. Les exigences de mise en œuvre sont donc par nécessité éparpillées dans la spécification. Les algorithmes ci-dessous (dans cette section) décrivent le cœur de ce traitement, mais ces algorithmes référencent et sont référencés par les règles d'analyse des balises de [script début](#) et [de fin](#) en HTML, [en contenu étranger](#) et [en XML](#), les règles de la [document.write\(\)](#) méthode, le traitement de [scénarisation](#), etc.

Lorsqu'ils sont insérés à l'aide de la [document.write\(\)](#) méthode, [script](#) les éléments s'exécutent [généralement \(bloquant généralement l'exécution ultérieure du script ou l'analyse HTML\)](#). Lorsqu'ils sont insérés à l'aide des attributs [innerHTML](#) et [outerHTML](#), ils ne s'exécutent pas du tout.

L' [defer](#) attribut peut être spécifié même si l' [async](#) attribut est spécifié, pour que les navigateurs Web hérités qui prennent uniquement en charge [defer](#) (et non [async](#)) reviennent au [defer](#) comportement au lieu du comportement de blocage qui est la valeur par défaut.

L' [crossorigin](#) attribut est un [attribut de paramètres CORS](#). Pour [les scripts classiques](#), il contrôle si les informations d'erreur seront exposées lorsque le script est obtenu à partir d'autres [origines](#). Pour [les scripts de module](#), il contrôle le [mode d'identification](#) utilisé pour les requêtes cross-origin.

Contrairement [aux scripts classiques](#), [les scripts de module](#) nécessitent l'utilisation du [protocole CORS](#) pour la récupération cross-origin.

L' [integrity](#) attribut représente les [métadonnées d'intégrité](#) des demandes dont cet élément est responsable. La valeur est du texte. L' [integrity](#) attribut ne doit pas être spécifié lorsque l' [src](#) attribut n'est pas spécifié. [\[ISR\]](#)

L' [referrerpolicy](#) attribut est un [attribut de stratégie de référence](#). Son but est de définir la [politique de référence](#) utilisée lors [de la récupération](#) du script, ainsi que tous les scripts importés à partir de celui-ci. [\[POLITIQUE DE RÉFÉRENCE\]](#)

Exemple de [script](#) stratégie de référencement d'un élément utilisée lors de la récupération de scripts importés, mais pas d'autres sous-ressources :

```
<script referrerpolicy="origin">
  fetch('/api/data');    // not fetched with <script>'s referrer
  policy
  import('./utils.mjs'); // is fetched with <script>'s referrer
  policy ("origin" in this case)
</script>
```

L' [blocking](#) attribut est un [attribut bloquant](#).

La modification dynamique des attributs [src](#), [type](#), [nomodule](#), [async](#), [defer](#), [crossorigin](#), [integrity](#) et [refe](#)

`referrerPolicy` n'a aucun effet direct ; ces attributs ne sont utilisés qu'à des moments spécifiques décrits ci-dessous.

Les attributs IDL `src`, `type`, `defer`, `integrity` et `blocking`, doivent chacun [réfléter](#) les attributs de contenu respectifs du même nom.



L' `referrerPolicy` attribut IDL doit [réfléter](#) l' `referrerPolicy` attribut contenu, [limité aux seules valeurs connues](#) .

L' `crossOrigin` attribut IDL doit [réfléter](#) l' `crossOrigin` attribut contenu, [limité aux seules valeurs connues](#) .

L' `noModule` attribut IDL doit [réfléter](#) l' `noModule` attribut contenu.

Les `async` étapes du getter sont :

1. Si [cette](#) force [asynchrone](#) est vraie, alors retournez true.
2. Si [cet](#) attribut de `async` contenu est présent, alors retourne true.
3. Renvoie faux.

Les `async` étapes du setter sont :

1. Définissez [cette](#) force [asynchrone](#) sur false.
2. Si la valeur donnée est true, définissez [cet](#) attribut de `async` contenu sur la chaîne vide.
3. Sinon, supprimez [cet](#) attribut de `async` contenu.

`script.text` [ = value ]

Renvoie le [contenu du texte enfant](#) de l'élément.

Peut être défini pour remplacer les enfants de l'élément par la valeur donnée.

`HTMLScriptElement.supports` ( type )

Renvoie true si le `type` donné est un type de script pris en charge par l'agent utilisateur. Les types de scripts possibles dans cette spécification sont " `classic`, " `module` " et " `importmap` ", mais d'autres pourraient être ajoutés à l'avenir.

Le `text` getter de l'attribut doit renvoyer [le contenu textuel enfant](#) `script` de cet élément .

Le `text` setter de l'attribut doit [chaîne remplacer all](#) par la valeur donnée dans cet `script` élément.

Les étapes de la méthode sont : `supports (type)`

1. Si `type` est `"classic"`, alors retourne vrai.
2. Si `type` est `"module"`, alors retourne vrai.
3. Si `type` est `"importmap"`, alors retourne vrai.
4. Renvoie faux.

*L'argument `type` doit correspondre exactement à ces valeurs ; nous n'effectuons pas de correspondance ASCII insensible à la casse . Ceci est différent de la façon dont type les valeurs d'attribut de contenu sont traitées et du fonctionnement DOMTokenList de la supports () méthode , mais cela correspond à l'WorkerType énumération utilisée dans le Worker () constructeur.*

Dans cet exemple, deux script éléments sont utilisés. L'un intègre un script classique externe et l'autre inclut des données sous forme de bloc de données .

```
<script src="game-engine.js"></script>
<script type="text/x-game-map">
.....U.....e
o.....A.....e
.....A.....AAA.....e
.A..AAA...AAAAA...e
</script>
```

Les données dans ce cas pourraient être utilisées par le script pour générer la carte d'un jeu vidéo. Les données ne doivent pas nécessairement être utilisées de cette façon ; peut-être que les données cartographiques sont en fait intégrées dans d'autres parties du balisage de la page, et le bloc de données ici est simplement utilisé par le moteur de recherche du site pour aider les utilisateurs qui recherchent des fonctionnalités particulières dans leurs cartes de jeu.

L'exemple suivant montre comment un script élément peut être utilisé pour définir une fonction qui est ensuite utilisée par d'autres parties du document, dans le cadre d'un script classique . Il montre également comment un script élément peut être utilisé pour invoquer un script pendant l'analyse du document, dans ce cas pour initialiser la sortie du formulaire.

```
<script>
function calculate(form) {
  var price = 52000;
  if (form.elements.brakes.checked)
    price += 1000;
  if (form.elements.radio.checked)
```

```

    price += 2500;
    if (form.elements.turbo.checked)
        price += 5000;
    if (form.elements.sticker.checked)
        price += 250;
    form.elements.result.value = price;
}
</script>
<form name="pricercalc" onsubmit="return false"
onchange="calculate(this)">
    <fieldset>
        <legend>Work out the price of your car</legend>
        <p>Base cost: £52000.</p>
        <p>Select additional options:</p>
        <ul>
            <li><label><input type=checkbox name=brakes> Ceramic brakes
(£1000)</label></li>
            <li><label><input type=checkbox name=radio> Satelllite radio
(£2500)</label></li>
            <li><label><input type=checkbox name=turbo> Turbo charger
(£5000)</label></li>
            <li><label><input type=checkbox name=sticker> "XZ" sticker
(£250)</label></li>
        </ul>
        <p>Total: £<output name=result></output></p>
    </fieldset>
    <script>
        calculate(document.forms.pricercalc);
    </script>
</form>

```

L'exemple suivant montre comment un [script](#)élément peut être utilisé pour inclure un [script de module JavaScript](#) externe .

```

<script type="module" src="app.mjs"></script>

```

Ce module, et toutes ses dépendances (exprimées par `import` des instructions JavaScript dans le fichier source), seront récupérés. Une fois que l'ensemble du graphe de module résultant a été importé et que le document a terminé l'analyse, le contenu de `app.mjs` sera évalué.

De plus, si le code d'un autre [script](#)élément du même [Window](#) importe le module depuis `app.mjs` (par exemple via `import './app.mjs';`), alors le même [script de module JavaScript](#) créé par l'ancien [script](#)élément sera importé.

Cet exemple montre comment inclure un [script de module JavaScript](#) pour les agents utilisateurs modernes et un [script classique](#) pour les agents utilisateurs plus anciens :

```
<script type="module" src="app.mjs"></script>
<script nomodule defer src="classic-app-bundle.js"></script>
```

Dans les agents utilisateurs modernes qui prennent en charge [les scripts de module JavaScript](#), l'[script](#)élément avec l'[nomodule](#)attribut sera ignoré et l'[script](#)élément avec un [type](#)" module" sera récupéré et évalué (en tant que [script de module JavaScript](#)). À l'inverse, les agents utilisateurs plus anciens ignoreront l'[script](#)élément avec un [type](#)" module", car il s'agit d'un type de script inconnu pour eux — mais ils n'auront aucun problème à récupérer et à évaluer l'autre [script](#)élément (comme un [script classique](#)), puisqu'ils n'implémentent pas le [nomodule](#)attribut.

L'exemple suivant montre comment un [script](#)élément peut être utilisé pour écrire un [script de module JavaScript](#) en ligne qui effectue un certain nombre de substitutions sur le texte du document, afin de rendre l'expérience de lecture plus intéressante (par exemple sur un site d'actualités) : [XKCD1288 ]

```
<script type="module">
  import { walkAllTextNodeDescendants } from "../dom-utils.mjs";

  const substitutions = new Map([
    ["witnesses", "these dudes I know"],
    ["allegedly", "kinda probably"],
    ["new study", "Tumblr post"],
    ["rebuild", "avenge"],
    ["space", "spaaace"],
    ["Google glass", "Virtual Boy"],
    ["smartphone", "Pokédex"],
    ["electric", "atomic"],
    ["Senator", "Elf-Lord"],
    ["car", "cat"],
    ["election", "eating contest"],
    ["Congressional leaders", "river spirits"],
    ["homeland security", "Homestar Runner"],
    ["could not be reached for comment", "is guilty and everyone knows it"]
  ]);

  function substitute(textNode) {
    for (const [before, after] of substitutions.entries()) {
      textNode.data = textNode.data.replace(new
      RegExp(`\\b${before}\\b`, "ig"), after);
    }
  }
}
```



```

    }
  }

  walkAllTextNodeDescendants(document.body, substitute);
</script>

```

Certaines fonctionnalités notables acquises en utilisant un script de module JavaScript incluent la possibilité d'importer des fonctions à partir d'autres modules JavaScript, le mode strict par défaut, et la façon dont les déclarations de niveau supérieur n'introduisent pas de nouvelles propriétés sur l'objet [global](#) . Notez également que, quel que soit l'endroit où cet [script](#) élément apparaît dans le document, il ne sera pas évalué tant que l'analyse du document ne sera pas terminée et que sa dépendance ( `dom-utils.mjs` ) n'aura pas été récupérée et évaluée.

L'exemple suivant montre comment un [script de module JSON](#) peut être importé depuis un [script de module JavaScript](#) :

```

<script type="module">
  import peopleInSpace from "http://api.open-notify.org/astros.json"
  assert { type: "json" };

  const list = document.querySelector("#people-in-space");
  for (const { craft, name } of peopleInSpace.people) {
    const li = document.createElement("li");
    li.textContent = `${name} / ${craft}`;
    list.append(li);
  }
</script>

```

La vérification du type MIME pour les scripts de module est stricte. Pour que la récupération du [script du module JSON](#) réussisse, la réponse HTTP doit avoir un [type JSON MIME](#) , par exemple `Content-Type: text/json`. D'autre part, si la `assert { type: "json" }` partie de l'instruction est omise, il est supposé que l'intention est d'importer un [script de module JavaScript](#) et la récupération échouera si la réponse HTTP a un type MIME qui n'est pas un [type MIME JavaScript](#) .

#### 4.12.1.1 Modèle de traitement

Un [script](#) élément a plusieurs états associés.

Un [script](#) élément a un **analyseur document** , qui est soit null soit a [Document](#), initialement null. Il est défini par l' [analyseur HTML](#) et l' [analyseur XML](#) sur [script](#) les éléments qu'ils insèrent et affecte le traitement de ces

éléments. [les éléments avec des documents d'analyseur](#)[script](#) non nuls sont appelés **parser-inserted** .

Un [script](#)élément a un **document de temps de préparation** , qui est soit nul, soit un [Document](#), initialement nul. Il est utilisé pour empêcher les scripts qui se déplacent entre les documents pendant [la préparation](#) de [s'exécuter](#) .

Un [script](#)élément a une **force** booléenne asynchrone, initialement vraie. Il est défini sur false par l' [analyseur HTML](#) et l' [analyseur XML](#) sur [script](#) les éléments qu'ils insèrent et lorsque l'élément reçoit un [async](#)attribut de contenu ajouté.

Un [script](#)élément a un booléen **provenant d'un fichier externe** , initialement faux. Il est déterminé lorsque le script est [préparé](#) , en fonction de l' [src](#) attribut de l'élément à ce moment-là.

Un [script](#)élément a un booléen **prêt à être exécuté par l'analyseur** , initialement faux. Ceci est utilisé uniquement pour les éléments qui sont également [insérés par l'analyseur](#) , pour que l'analyseur sache quand exécuter le script.

Un [script](#)élément a un booléen **déjà commencé** , initialement faux.

Un [script](#)élément a un booléen **retardant l'événement de chargement** , initialement faux.

Un [script](#)élément a un **type** , qui est soit null, " `classic`", " `module`" ou " `importmap`", initialement null. Il est déterminé lorsque l'élément est [préparé](#) , en fonction de l' [type](#)attribut de l'élément à ce moment-là.

Un [script](#)élément a un **résultat** qui est soit " `uninitialized`", null (représentant une erreur), un [script](#) ou un [résultat d'analyse de carte d'importation](#) . C'est initialement " `uninitialized`".

Un [script](#)élément a **des étapes à exécuter lorsque le résultat est prêt** , qui sont une série d'étapes ou null, initialement null. Pour **marquer comme prêt** un [script](#)élément *e/* étant donné un [script](#) , [importez le résultat de l'analyse de la carte](#) ou *un résultat* nul :

1. Définissez [le résultat](#) de *e/* sur *result* .
2. Si les étapes de *e/* [à exécuter lorsque le résultat est prêt](#) ne sont pas nulles, alors exécutez-les.
3. Définissez les étapes de *e/* [pour qu'elles s'exécutent lorsque le résultat est prêt](#) à être nul.
4. Définissez *e/* retardant [l'événement de chargement](#) sur false.

---

Un script élément *e/* est implicitement potentiellement bloquant le rendu si le type de *e/* est " `classic`", *e/* est inséré par l'analyseur et *e/* n'a pas d'attribut async ou defer

Les étapes de clonage d'un script élément *e/* en cours de clonage vers une *copie* consistant à définir les copies déjà commencées sur *e/* déjà commencées .

Lorsqu'un async attribut est ajouté à un script élément *e/* , l'agent utilisateur doit définir force async de *e/* sur `false`.

Chaque fois qu'un script élément *e/* retarde l'événement load est vrai, l'agent utilisateur doit retarder l'événement load du document de temps de préparation de *e/* .

---

Lorsqu'un script élément *e/* qui n'est pas inséré par l'analyseur subit l'un des événements répertoriés dans la liste suivante, l'agent utilisateur doit immédiatement préparer l'élément de script *e/* :

- L' script élément devient connecté .
- L' script élément est connecté et un nœud ou un fragment de document est inséré dans l' script élément, après tous les script éléments insérés à ce moment-là.
- L' script élément est connecté et possède un src ensemble d'attributs là où auparavant l'élément n'avait pas un tel attribut.

Pour **préparer l'élément de script** donné un script élément *e/* :

1. Si *e/* est déjà commencé est vrai, alors retournez.
2. Soit *parser document* le document parser de *e/* .
3. Définissez le document d'analyseur de *e/* sur `null`.

*Ceci est fait pour que si script les éléments insérés par l'analyseur échouent à s'exécuter lorsque l'analyseur essaie de les exécuter, par exemple parce qu'ils sont vides ou spécifient un langage de script non pris en charge, un autre script peut ensuite les muter et les faire s'exécuter à nouveau.*

4. Si le document de l'analyseur est non nul et que *el* n'a pas d' asyncattribut, alors définissez la force asynchrone de *el* sur true.

*Ceci est fait de sorte que si un scriptélément inséré par l'analyseur ne s'exécute pas lorsque l'analyseur tente de l'exécuter, mais qu'il est exécuté ultérieurement après qu'un script l'ait mis à jour dynamiquement, il s'exécutera de manière asynchrone même si l' asyncattribut n'est pas défini.*

5. Soit le texte source le contenu du texte enfant de *el* .
6. Si *el* n'a pas srcd'attribut et que le texte source est la chaîne vide, alors retournez.
7. Si *el* n'est pas connecté , alors retour.
8. Si l'une des conditions suivantes est vraie :
- *el* a un typeattribut dont la valeur est la chaîne vide ;
  - *el* n'a pas typedd'attribut mais il a un languageattribut et la valeur de cet attribut est la chaîne vide ; ou
  - *el* n'a ni typeattribut ni languageattribut

puis laissez la chaîne de type du bloc de script pour cet scriptélément être "text/javascript".

Sinon, si *el* a un typeattribut, laissez la chaîne de type du bloc de script être la valeur de cet attribut avec les espaces blancs ASCII de début et de fin supprimés .

Sinon, *el* a un attribut non vide language ; laissez la chaîne de type du bloc de script être la concaténation de "text/" et la valeur de l'attribut *el*language .

*L' languageattribut n'est jamais conforme et est toujours ignoré si un type attribut est présent.*

9. Si la chaîne de type du bloc de script est une correspondance d'essence de type JavaScript MIME , alors définissez le type de *el* sur ".classic"
10. Sinon, si la chaîne de type du bloc de script est une correspondance ASCII insensible à la casse pour la chaîne " module", alors définissez le type de *el* sur ".module"
11. Sinon, si la chaîne de type du bloc de script est une correspondance ASCII insensible à la casse pour la chaîne " importmap", alors définissez le type de *el* sur ".importmap"
12. Sinon, reviens. (Aucun script n'est exécuté et le type de *el* reste nul.)

13. Si *l'analyseur de document* n'est pas nul, redéfinissez [l'analyseur de document](#) de *el* sur *l'analyseur de document* et définissez [la force asynchrone](#) de *el* sur *false*.
14. Set *el* a [déjà commencé](#) à *true*.
15. Définissez [le document de temps de préparation](#) de *el* sur son [nœud document](#) .
16. Si *parser document* n'est pas nul et que *parser document* n'est pas égal au [document de temps de préparation](#) de *el* , alors retournez.
17. Si [le script est désactivé](#) pour *el* , alors retournez.

*La définition de [script désactivé](#) signifie que, entre autres, les scripts suivants ne s'exécuteront pas : les scripts dans les documents [XMLHttpRequest](#) de , les scripts dans les documents créés par -, les scripts dans les documents créés par la fonctionnalité de et les scripts qui sont d'abord insérés par un script dans *a* qui a été créé à l'aide de l' API. [\[XHR\]](#) [\[DOMPARSING\]](#) [\[XSLTP\]](#) [\[DOM\]](#) [responseXMLDOMParserXSLTProcessortransformToDocumentDocumentcreateDocument\(\)](#)*

18. Si *el* a un [nomodule](#) attribut de contenu et que son [type](#) est " *classic* ", alors retournez.

*Cela signifie que spécifier [nomodule](#) sur un [script de module](#) n'a aucun effet ; l'algorithme continue.*

19. Si *el* n'a pas d' [src](#) attribut de contenu, et le [comportement en ligne de l'élément doit-il être bloqué par la politique de sécurité du contenu ?](#) algorithm renvoie " *Blocked* " lorsqu'il est donné *el* , " *script* " et *source text* , puis return. [\[CSP\]](#)
20. Si *el* a un [event](#) attribut et un [for](#) attribut, et que [le type](#) de *el* est " ", alors :*classic*
  - Soit la valeur de l' attribut *el* 's' [for](#).
  - Soit *event* la valeur de l' attribut *el* [event](#) .
  - [Supprimez les espaces blancs ASCII de début et de fin](#) de l'événement et *pour* .
  - Si *for* n'est pas une correspondance [ASCII insensible à la casse](#) pour la chaîne " *window* ", alors retournez.
  - Si l'événement n'est pas une correspondance [ASCII insensible à la casse](#) pour la chaîne " *onload* " ou la chaîne " *onload()* ", alors retour.
21. Si *el* a un [charset](#) attribut, alors laissez *encoding* être le résultat de [l'obtention d'un encodage](#) à partir de la valeur de l' [charset](#) attribut.

Si *el* n'a pas d' [charset](#) attribut, ou si [l'obtention d'un codage](#) a échoué, alors laissez *encoding* être le nœud de *el* [document](#) 's [the encoding](#) .

*Si le type de el est " ", cet encodage sera ignoré.*[module](#)

22. Soit *le paramètre CORS* du script classique l'état actuel de l'attribut content de *el*[crossorigin](#) .
23. Soit *le mode d'informations d'identification* du script de module le [mode d'informations d'identification de l'attribut de paramètres CORS](#) pour l'attribut de contenu de *el*[crossorigin](#) .
24. Soit *nonce cryptographique* la valeur de l'emplacement interne [\[\[CryptographicNonce\]\]](#) de *el* .
25. Si *el* a un [integrity](#) attribut, alors laissez *les métadonnées d'intégrité* être la valeur de cet attribut.  
  
Sinon, laissez *les métadonnées d'intégrité* être la chaîne vide.
26. Soit *referrer policy* l'état actuel de l'attribut content de *el*[referrerpolicy](#) .
27. Laissez *les métadonnées de l'analyseur* être " `parser-inserted`" si *el* est [inséré par l'analyseur](#) , et " `not-parser-inserted`" sinon.
28. Soit *options* une [option de récupération de script](#) dont [le nonce cryptographique](#) est *le nonce cryptographique* , [les métadonnées d'intégrité](#) sont *les métadonnées d'intégrité* , [les métadonnées de l'analyseur](#) sont *les métadonnées de l'analyseur* , [le mode d'identification](#) est *le mode d'identification du script de module* et [la politique de référence](#) est *la politique de référence* .
29. Soit *l' objet settings* être [l' objet de paramètres pertinent](#) du document du [nœud](#) *el* .
30. Si *el* a un [src](#) attribut de contenu, alors :

- Si le [type](#) de *el* est " ", alors [mettez en file d'attente une tâche d'élément](#) sur la [source de la tâche de manipulation DOM](#) donnée à *el* pour [déclencher un événement](#) nommé à *el* , et retournez `importmap`[error](#)

*Les scripts de carte d'importation externe ne sont actuellement pas pris en charge. Voir [WICG/import-maps issue #235](#) pour des discussions sur l'ajout de support.*

- Soit *src* la valeur de l' attribut *el*[src](#) .
- Si *src* est la chaîne vide, alors [mettez en file d'attente une tâche d'élément](#) sur la [source de tâche de manipulation](#)

DOM donnée *el* pour déclencher un événement nommé error à *el*, et retour.

- Définissez *el* à partir d'un fichier externe sur *true*.
- Parse *src* par rapport au document de noeud d' *el*.
- Si l'étape précédente a échoué, mettez en file d'attente une tâche d'élément sur la source de la tâche de manipulation DOM donnée à *el* pour déclencher un événement nommé error à *el*, et revenez. Sinon, laissez *url* être l' enregistrement d'URL résultant.
- Si *el* bloque potentiellement le rendu, alors bloquez le rendu sur *el*.
- Définissez *el* retardant l'événement de chargement sur *true*.
- Si *el* est actuellement en train de bloquer le rendu, alors définissez le blocage du rendu des *options* sur *true*.
- Soit *onComplete* le résultat donné soit les étapes suivantes :
  1. Marquer comme prêt le résultat donné.
- Activez le type de *el*:

```
" classic"
```

Récupérez un script classique avec l'*URL*, l'*objet de paramètres*, les *options*, le paramètre *CORS* du script classique, l'encodage et *onComplete*.

```
" module"
```

Récupérez un graphique de script de module externe donné *url*, *settings* *object*, *options* et *onComplete*.

Pour des raisons de performances, les agents utilisateurs peuvent commencer à récupérer le script classique ou le graphe de module (tel que défini ci-dessus) dès que l' srcattribut est défini, à la place, dans l'espoir que *el* sera inséré dans le document (et que l' crossoriginattribut ne changera pas valeur entre-temps). Dans tous les cas, une fois *el* inséré dans le document, le chargement doit avoir commencé comme décrit dans cette étape. Si l'UA effectue une telle prélecture, mais *el* n'est jamais inséré dans le document, ou l' srcattribut est modifié dynamiquement, ou le crossoriginest modifié dynamiquement, l'agent utilisateur n'exécutera pas le script ainsi obtenu et le processus de récupération aura été effectivement gaspillé.

31. Si *el* n'a pas d' srcattribut content :

- Soit l' *URL de base* l' URL de base du document du noeud *el*.
- Activez le type de *el*:



" classic"

1. Soit *script* le résultat de [la création d'un script classique](#) à l'aide *du texte source* , *de l'objet de paramètres* , *de l'URL de base* et *des options* .
2. [Marquer comme prêt](#) le script donné .

" module"

3. Définissez *el* retardant [l'événement de chargement](#) sur *true*.
4. [Récupérez un graphique de script de module en ligne](#) , *le texte source* donné , *l'URL de base* , *l'objet de paramètres* , *les options* , et avec les étapes suivantes, *le résultat* est donné :

1. [Marquer comme prêt](#) le résultat donné .

" importmap"

5. Si les cartes d'importation de l' [objet global pertinent](#) de *el* [autorisées](#) sont fausses, alors [mettez en file d'attente une tâche d'élément](#) sur la [source de la tâche de manipulation DOM](#) donnée à *el* pour [déclencher un événement](#) nommé à *el* , et revenez.[error](#)
6. Définissez les cartes d'importation de l' objet [global pertinent](#) de *el* [autorisées](#) sur *false*.
7. Soit *result* le résultat de [la création d'un résultat d'analyse de carte d'importation](#) en fonction *du texte source* et *de l'URL de base* .
8. [Marquer comme prêt](#) le résultat donné .

32. Si *le type* de *el* est " classic" et *que el* a un [src](#) attribut, ou si *le type* de *el* est " module" :

- [Assert](#) : [le résultat](#) de *el* est " ".uninitialized
- Si *el* a un [async](#) attribut ou si [la force asynchrone](#) de *el* est vraie :
  1. Soit *scripts* l' ensemble de scripts *du document* [de préparation de](#) *el* [qui s'exécutera dès que possible](#) .
  2. [Ajouter](#) *el* aux *scripts* .
  3. Définissez les étapes de *el* [pour qu'elles s'exécutent lorsque le résultat est prêt](#) :
    1. [Exécutez l'élément de script](#) *el* .
    2. [Supprimer](#) *el* des *scripts* .
- Sinon, si *el* n'est pas [inséré par l'analyseur](#) :



1. Soit *scripts* la liste des scripts du document de préparation de *el* qui s'exécuteront dans l'ordre dès que possible .
2. Ajouter *el* aux *scripts* .
3. Définissez les étapes de *el* pour qu'elles s'exécutent lorsque le résultat est prêt :
  1. Si *scripts* [0] n'est pas *el* , alors abandonnez ces étapes.
  2. Tant que *scripts* n'est pas vide et que le résultat de *scripts* [0] n'est pas " " :uninitialized
    1. Exécutez les *scripts* d'élément de script [0].
    2. Supprimer les *scripts* [0].
- Sinon, si *el* a un defer attribut ou si le type de *el* est " " :module
  1. Ajoutez *el* à la liste des scripts de son document analyseur qui s'exécuteront lorsque le document aura terminé l'analyse .
  2. Définissez les étapes de *el* pour qu'elles s'exécutent lorsque le résultat est prêt comme suit : définissez *el* prêt à être exécuté par l'analyseur sur true. (L'analyseur se chargera d'exécuter le script.)
- Sinon:
  1. Définissez le script de blocage d'analyse en attente du document d' analyseur *el* sur *el* .
  2. Bloquer le rendu sur *el* .
  3. Définissez les étapes de *el* pour qu'elles s'exécutent lorsque le résultat est prêt comme suit : définissez *el* prêt à être exécuté par l'analyseur sur true. (L'analyseur se chargera d'exécuter le script.)

33. Sinon:

- Assert : le résultat de *el* n'est pas " ".uninitialized
- Si toutes les conditions suivantes sont vraies :
  1. *le type* de *el* est " classic" ;
  2. *el* est inséré par l'analyseur ;
  3. le document d'analyseur d' *el* a une feuille de style qui bloque les scripts ; et
  4. soit l'analyseur qui a créé *el* est un analyseur XML , soit c'est un analyseur HTML dont le niveau d'imbrication de script n'est pas supérieur à un,

alors:

5. Définissez le script de blocage d'analyse en attente du document d' analyseur el sur el .
6. Définissez el/prêt à être exécuté par l'analyseur sur true. (L'analyseur se chargera d'exécuter le script.)
  - Sinon, exécutez immédiatement l'élément de script el , même si d'autres scripts sont déjà en cours d'exécution.

Chacun Documenta un **script de blocage d'analyse en attente** , qui est un scriptélément ou null, initialement null.

Chacun Documenta un **ensemble de scripts qui s'exécuteront dès que possible** , qui est un ensemble d' scriptéléments, initialement vides.

Chacun Documenta une **liste de scripts qui s'exécuteront dans l'ordre dès que possible** , qui est une liste d' scriptéléments, initialement vide.

Chacun Documenta une **liste de scripts qui s'exécuteront lorsque le document aura fini d'analyser** , qui est une liste d' scriptéléments, initialement vide.

*Si un scriptélément qui bloque un analyseur est déplacé vers un autre Document avant qu'il n'ait normalement cessé de bloquer cet analyseur, il continue néanmoins à bloquer cet analyseur jusqu'à ce que la condition qui l'a fait bloquer l'analyseur ne s'applique plus (par exemple, si le script est en attente script de blocage d'analyse car l'original Document a une feuille de style qui bloque les scripts lorsqu'il a été analysé, mais le script est ensuite déplacé vers un autre Document avant le chargement de la ou des feuilles de style de blocage, le script bloque toujours l'analyseur jusqu'à ce que les feuilles de style soient toutes chargées, moment auquel le script s'exécute et l'analyseur est débloqué).*

Pour **exécuter l'élément de script** donné un scriptélément el :

1. Soit *document* le nœud document de el .
2. Si le document de temps de préparation de el n'est pas égal à *document* , alors retournez.
3. Débloquer le rendu sur el .
4. Si le résultat de el est nul, déclenchez un événement nommé à el et retournez.error
5. Si el d' un fichier externe est vrai, ou si le type de el est " " , alors incrémente le compteur ignore-destructive-writes du *document* .`module`
6. Activez le type de el :

" classic"

1. Soit `oldCurrentScript` la valeur à laquelle l'objet `document``currentScript` a été défini le plus récemment.
2. Si [la racine](#) de `el` n'est pas une [racine fantôme](#) , alors définissez l'attribut de `document` sur `el` . Sinon, réglez-le sur `null`.`currentScript`

*Cela n'utilise pas le [dans une](#) vérification d'arborescence de documents, car `el` aurait pu être supprimé du document avant l'exécution, et dans ce scénario, il `currentScript` doit toujours pointer vers lui.*

3. [Exécutez le script classique](#) donné par [le résultat](#) de `el` .
4. Définissez l'attribut du `document``currentScript` sur `oldCurrentScript` .

" module"

5. [Assert](#) : l'attribut du `document``currentScript` est nul.
6. [Exécutez le script de module](#) donné par [le résultat](#) de `el` .

" importmap"

7. [Enregistrez une carte d'importation](#) en fonction de [l'objet global pertinent](#) de `el` et [du résultat](#) de `el` .
7. Décrémentez le [compteur ignore-destructive-writes](#) de `document` , s'il a été incrémenté à l'étape précédente.
8. Si `el` provient [d'un fichier externe](#) est vrai, [déclenchez un événement](#) nommé `load` à `el` .

#### 4.12.1.2 Langages de script

Les agents utilisateurs ne sont pas tenus de prendre en charge JavaScript. Cette norme doit être mise à jour si un langage autre que JavaScript arrive et obtient une large adoption similaire par les navigateurs Web. Jusqu'à ce moment, l'implémentation d'autres langages est en conflit avec cette norme, compte tenu du modèle de traitement défini pour l' `script` élément.

Les serveurs doivent utiliser `text/javascript` pour les ressources JavaScript, conformément à *Updates to ECMAScript Media Types* . Les serveurs ne doivent pas utiliser d'autres [types JavaScript MIME](#) pour les ressources JavaScript et ne doivent pas utiliser [de types MIME non JavaScript](#) . [\[RFC9239\]](#)

Pour les ressources JavaScript externes, les paramètres de type MIME dans [Content-Type](#) les en-têtes `` sont généralement ignorés. (Dans certains cas, le `charset` paramètre `` a un effet.) Cependant, pour l'attribut `script` de l'élément [type](#), ils sont significatifs ; il utilise le concept [de correspondance d'essence de type JavaScript MIME](#) .

*Par exemple, les scripts dont [type](#) l'attribut est défini sur "text/javascript; charset=utf-8" ne seront pas évalués, même s'il s'agit d'un [type MIME JavaScript](#) valide lors de l'analyse.*

De plus, toujours pour les ressources JavaScript externes, des considérations spéciales s'appliquent autour [Content-Type](#) du traitement de l'en-tête `` comme détaillé dans l'algorithme [de préparation de l'élément de script](#) et *Fetch* . [\[ALLER CHERCHER\]](#)

#### 4.12.1.3 Restrictions pour le contenu des `script` éléments

*Le moyen le plus simple et le plus sûr d'éviter les restrictions plutôt étranges décrites dans cette section est de toujours échapper une correspondance ASCII insensible à la casse pour " " <!-- as " \x3C!--", " <script" as " \x3Cscript" et " " </script as " \x3C/script" lorsque ces séquences apparaissent dans des littéraux dans des scripts (par exemple dans des chaînes, des expressions régulières ou des commentaires) et pour éviter d'écrire du code qui utilise de telles constructions dans des expressions. Cela évite les pièges que les restrictions de cette section sont susceptibles de déclencher : à savoir que, pour des raisons historiques, l'analyse de [script](#) blocs en HTML est une pratique étrange et exotique qui agit de manière non intuitive face à ces séquences.*

Le [contenu](#)`script` du texte descendant de l'élément doit correspondre à la production dans l'ABNF suivant, dont le jeu de caractères est Unicode. [\[ABNF\]](#)`script`

```
script      = outer *(comment-open inner comment-close outer )
```

```
outer       = < any string that doesn't contain a substring that
matches not-in-outer >
```

```
not-in-outer = comment-open
```

```
inner       = < any string that doesn't contain a substring that
matches not-in-inner >
```

```
not-in-inner = comment-close / script-open
```

```
comment-open = "<!--"
```

```
comment-close = "-->"
```

```
script-open  = "<" s c r i p t tag-end
```

```

s      = %x0053 ; U+0053 LATIN CAPITAL LETTER S
s      =/ %x0073 ; U+0073 LATIN SMALL LETTER S
c      = %x0043 ; U+0043 LATIN CAPITAL LETTER C
c      =/ %x0063 ; U+0063 LATIN SMALL LETTER C
r      = %x0052 ; U+0052 LATIN CAPITAL LETTER R
r      =/ %x0072 ; U+0072 LATIN SMALL LETTER R
i      = %x0049 ; U+0049 LATIN CAPITAL LETTER I
i      =/ %x0069 ; U+0069 LATIN SMALL LETTER I
p      = %x0050 ; U+0050 LATIN CAPITAL LETTER P
p      =/ %x0070 ; U+0070 LATIN SMALL LETTER P
t      = %x0054 ; U+0054 LATIN CAPITAL LETTER T
t      =/ %x0074 ; U+0074 LATIN SMALL LETTER T

tag-end = %x0009 ; U+0009 CHARACTER TABULATION (tab)
tag-end =/ %x000A ; U+000A LINE FEED (LF)
tag-end =/ %x000C ; U+000C FORM FEED (FF)
tag-end =/ %x0020 ; U+0020 SPACE
tag-end =/ %x002F ; U+002F SOLIDUS (/)
tag-end =/ %x003E ; U+003E GREATER-THAN SIGN (>)

```

Lorsqu'un script élément contient [une documentation de script](#) , il existe d'autres restrictions sur le contenu de l'élément, comme décrit dans la section ci-dessous.

Le script suivant illustre ce problème. Supposons que vous ayez un script contenant une chaîne, comme dans :

```

const example = 'Consider this string: <!-- <script>';
console.log(example);

```

Si l'on mettait cette chaîne directement dans un script bloc, cela violerait les restrictions ci-dessus :

```

<script>
  const example = 'Consider this string: <!-- <script>';
  console.log(example);
</script>

```

Le plus gros problème, cependant, et la raison pour laquelle cela violerait ces restrictions, est qu'en fait le script serait analysé bizarrement : *le bloc de script ci-dessus n'est pas terminé* . Autrement dit, ce qui ressemble à une `</script>` balise de fin " " dans cet extrait fait toujours partie du script bloc. Le script ne s'exécute pas (puisqu'il n'est pas terminé) ; s'il devait s'exécuter d'une manière ou d'une autre,

comme cela pourrait être le cas si le balisage ressemblait à ce qui suit, il échouerait car le script (mis en évidence ici) n'est pas un JavaScript valide :

```
<script>
  const example = 'Consider this string: <!-- <script>';
  console.log(example);
</script>

<!-- despite appearances, this is actually part of the script
still! -->

<script>
  ... // this is the same script block still...
</script>
```

Ce qui se passe ici, c'est que pour des raisons d'héritage, les chaînes "<!--" et "<script" dans les éléments HTML doivent être équilibrées pour que l'analyseur envisage de fermer le bloc. <scriptscript

En échappant aux chaînes problématiques comme mentionné en haut de cette section, le problème est entièrement évité :

```
<script>
  // Note: `&lt;` is an escape sequence for `&lt;`.
  const example = 'Consider this string: &lt;!-- &lt;script>';
  console.log(example);
</script>

<!-- this is just a comment between script blocks -->

<script>
  ... // this is a new script block
</script>
```

Il est possible que ces séquences apparaissent naturellement dans les expressions de script, comme dans les exemples suivants :

```
if (x<!--y) { ... }
if ( player<script ) { ... }
```

Dans de tels cas, les caractères ne peuvent pas être échappés, mais les expressions peuvent être réécrites afin que les séquences ne se produisent pas, comme dans :

```
if (x < !--y) { ... }
if (!--y > x) { ... }
if (!(--y) > x) { ... }
if (player < script) { ... }
if (script > player) { ... }
```

Cela évite également un écueil différent : pour des raisons historiques connexes, la chaîne "<!--" dans [les scripts classiques](#) est en fait traitée comme un début de commentaire de ligne, tout comme "/\*".

#### 4.12.1.4 Documentation en ligne pour les scripts externes

Si l'attribut [script](#) d'un élément [src](#) est spécifié, alors le contenu de l'[script](#) élément, le cas échéant, doit être tel que la valeur de l'[text](#) attribut IDL, qui est dérivé du contenu de l'élément, corresponde à la `documentation` production dans l'ABNF suivant, dont le jeu de caractères est Unicode. [\[ABNF\]](#)

```
documentation = *( *( space / tab / comment ) [ line-comment ]
newline )

comment       = slash star *( not-star / star not-slash ) 1*star
slash

line-comment  = slash slash *not-newline

; characters

tab           = %x0009 ; U+0009 CHARACTER TABULATION (tab)
newline       = %x000A ; U+000A LINE FEED (LF)
space         = %x0020 ; U+0020 SPACE
star          = %x002A ; U+002A ASTERISK (*)
slash         = %x002F ; U+002F SOLIDUS (/)
not-newline   = %x0000-0009 / %x000B-10FFFF
               ; a scalar value other than U+000A LINE FEED (LF)
not-star      = %x0000-0029 / %x002B-10FFFF
               ; a scalar value other than U+002A ASTERISK (*)
not-slash     = %x0000-002E / %x0030-10FFFF
               ; a scalar value other than U+002F SOLIDUS (/)
```

*Cela correspond à mettre le contenu de l'élément dans les commentaires JavaScript. Cette exigence s'ajoute aux restrictions précédentes sur la syntaxe du contenu des [script](#) éléments.*

Cela permet aux auteurs d'inclure de la documentation, telle que des informations de licence ou des informations sur l'API, dans leurs documents tout en se référant à des fichiers de script externes. La syntaxe est contrainte afin que les auteurs n'incluent pas accidentellement ce qui ressemble à un script valide tout en fournissant un [src](#) attribut.

```
<script src="cool-effects.js">
  // create new instances using:
  //   var e = new Effect();
  // start the effect using .play, stop using .stop:
```

```
//      e.play();  
//      e.stop();  
</script>
```

#### 4.12.1.5 Interaction des script éléments et XSLT

*Cette section est non normative.*

Cette spécification ne définit pas comment XSLT interagit avec l' script élément. Cependant, en l'absence d'une autre spécification définissant réellement cela, voici quelques lignes directrices pour les implémenteurs, basées sur les implémentations existantes :

- Lorsqu'un programme de transformation XSLT est déclenché par une `<?xml-stylesheet?>` instruction de traitement et que le navigateur implémente une transformation directe vers DOM, script les éléments créés par le processeur XSLT doivent avoir leur document d'analyse correctement défini et exécutés dans l'ordre du document (scripts modulo marqués defer ou async), immédiatement, au fur et à mesure que la transformation se produit.
- La `XSLTProcessor.transformToDocument()` méthode ajoute des éléments à un Document objet avec un contexte de navigation nul et, par conséquent, tous script les éléments qu'ils créent doivent avoir leur déjà commencé défini sur `true` dans l' algorithme de préparation de l'élément de script et ne jamais être exécutés ( le script est désactivé ). Cependant, ces script éléments doivent toujours avoir leur ensemble de documents d'analyseur, de sorte que leur async attribut IDL renverra faux en l'absence d'un async attribut de contenu.
- La `XSLTProcessor.transformToFragment()` méthode doit créer un fragment équivalent à celui construit manuellement en créant les éléments à l'aide de `document.createElementNS()`. Par exemple, il doit créer script des éléments avec un document d'analyseur nul et avec leur déjà commencé défini sur `false`, afin qu'ils s'exécutent lorsque le fragment est inséré dans un document.

La principale distinction entre les deux premiers cas et le dernier cas est que les deux premiers opèrent sur Documents et le dernier opère sur un fragment.

#### 4.12.2 L' noscript élément



### Catégories :

[Contenu des métadonnées](#) .  
[Contenu du flux](#) .  
[Contenu de la phrase](#) .

### Contextes dans lesquels cet élément peut être utilisé :

Dans un [head](#)élément d'un [document HTML](#) , s'il n'y a pas [noscript](#)d'éléments ancêtres.  
Où [le contenu de phrasé](#) est attendu dans [les documents HTML](#) , s'il n'y a pas [noscript](#)d'éléments ancêtres.

### Modèle de contenu :

Lorsque [le script est désactivé](#) , dans un [head](#)élément : dans n'importe quel ordre, zéro ou plusieurs [link](#)éléments, zéro ou plusieurs [style](#)éléments, et zéro ou plusieurs [meta](#)éléments.  
Lorsque [le script est désactivé](#) , pas dans un [head](#)élément : [transparent](#) , mais il ne doit y avoir aucun [noscript](#)descendant d'élément.  
Sinon : texte conforme aux exigences données dans la prose.

### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

### Attributs de contenu :

[Attributs globaux](#)

### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

### Interface DOM :

Utilisations [HTML](#)[Element](#).

L' [noscript](#)élément ne [représente](#) rien si [le script est activé](#) et [représente](#) ses enfants si [le script est désactivé](#) . Il est utilisé pour présenter un balisage différent aux agents utilisateurs qui prennent en charge les scripts et à ceux qui ne prennent pas en charge les scripts, en affectant la façon dont le document est analysé.

Lorsqu'il est utilisé dans [des documents HTML](#) , le modèle de contenu autorisé est le suivant :

Dans un [head](#)élément, si [le script est désactivé](#) pour l' [noscript](#)élément

L' [noscript](#)élément doit contenir uniquement les éléments [link](#), [style](#)et [meta](#).

Dans un [head](#)élément, si [le script est activé](#) pour l' [noscript](#)élément

L' [noscript](#)élément ne doit contenir que du texte, sauf que l'appel de l' [algorithme d'analyse de fragment HTML](#) avec l' [noscript](#)élément comme élément de [contexte](#) et le contenu du texte comme *entrée* doit aboutir à une liste de nœuds composée uniquement d' éléments [link](#), [style](#)et [meta](#)qui

seraient conformes s'ils étaient des enfants de l' [noscript](#)élément, et aucune [erreur d'analyse](#) .

**En dehors des [head](#)éléments, si [le script est désactivé](#) pour l' [noscript](#)élément**

Le [noscript](#)modèle de contenu de l'élément est [transparent](#) , avec la restriction supplémentaire qu'un [noscript](#)élément ne doit pas avoir d' [noscript](#) élément comme ancêtre (c'est-à-dire [noscript](#)qu'il ne peut pas être imbriqué).

**En dehors des [head](#)éléments, si [le script est activé](#) pour l' [noscript](#)élément**

L' [noscript](#)élément ne doit contenir que du texte, sauf que le texte doit être tel que l'exécution de l'algorithme suivant aboutit à un document conforme sans [noscript](#)éléments et sans [script](#)éléments, et tel qu'aucune étape de l'algorithme ne lève d'exception ou ne force un [analyseur HTML](#) à signaler un [erreur d'analyse](#) :

1. Supprimez chaque [script](#)élément du document.
2. Faites une liste de tous [noscript](#)les éléments du document. Pour chaque [noscript](#)élément de cette liste, procédez comme suit :
  1. Soit s le [contenu textuel enfant](#) de l' [noscript](#) élément.
  2. Définissez l' [outerHTML](#)attribut de l' [noscript](#)élément sur la valeur de s . (Ceci, comme effet secondaire, entraîne la [noscript](#)suppression de l'élément du document.) [\[DOMPARSING\]](#)

*Toutes ces contorsions sont nécessaires car, pour des raisons historiques, l' [noscript](#)élément est traité différemment par l' [analyseur HTML](#) selon que [le script a été activé ou non](#) lorsque l'analyseur a été appelé.*

L' [noscript](#)élément ne doit pas être utilisé dans [les documents XML](#) .

*L' [noscript](#)élément n'est effectif que dans [la syntaxe HTML](#) , il n'a aucun effet dans [la syntaxe XML](#) . En effet, cela fonctionne essentiellement en "désactivant" l'analyseur lorsque les scripts sont activés, de sorte que le contenu de l'élément est traité comme du texte pur et non comme des éléments réels. XML ne définit pas de mécanisme permettant de le faire.*

L' [noscript](#)élément n'a pas d'autres exigences. En particulier, les enfants de l' [noscript](#)élément ne sont pas exemptés de [soumission de formulaire](#) , de script, etc., même lorsque [le script est activé](#) pour l'élément.

Dans l'exemple suivant, un [noscript](#)élément est utilisé pour fournir une solution de secours à un script.

```

<form action="calcSquare.php">
  <p>
    <label for=x>Number</label>:
    <input id="x" name="x" type="number">
  </p>
  <script>
    var x = document.getElementById('x');
    var output = document.createElement('p');
    output.textContent = 'Type a number; it will be squared right
then!';
    x.form.appendChild(output);
    x.form.onsubmit = function () { return false; }
    x.oninput = function () {
      var v = x.valueAsNumber;
      output.textContent = v + ' squared is ' + v * v;
    };
  </script>
  <noscript>
    <input type=submit value="Calculate Square">
  </noscript>
</form>

```

Lorsque le script est désactivé, un bouton apparaît pour effectuer le calcul côté serveur. Lorsque le script est activé, la valeur est calculée à la volée à la place.

L' noscript élément est un instrument contondant. Parfois, les scripts peuvent être activés, mais pour une raison quelconque, le script de la page peut échouer. Pour cette raison, il est généralement préférable d'éviter d'utiliser noscript, et de plutôt concevoir le script pour faire passer la page d'une page sans script à une page avec script à la volée, comme dans l'exemple suivant :

```

<form action="calcSquare.php">
  <p>
    <label for=x>Number</label>:
    <input id="x" name="x" type="number">
  </p>
  <input id="submit" type=submit value="Calculate Square">
  <script>
    var x = document.getElementById('x');
    var output = document.createElement('p');
    output.textContent = 'Type a number; it will be squared right
then!';
    x.form.appendChild(output);

```

```

x.form.onsubmit = function () { return false; }
x.oninput = function () {
    var v = x.valueAsNumber;
    output.textContent = v + ' squared is ' + v * v;
};
var submit = document.getElementById('submit');
submit.parentNode.removeChild(submit);
</script>
</form>

```

La technique ci-dessus est également utile dans [les documents XML](#) , car [noscript](#) elle n'y est pas autorisée.

#### 4.12.3 L' [template](#) élément



##### Catégories :

[Contenu des métadonnées](#) .  
[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Élément de support de script](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu des métadonnées](#) est attendu.  
 Où [le contenu du phrasé](#) est attendu.  
 Où [les éléments de support de script](#) sont attendus.  
 En tant qu'enfant d'un [colgroup](#) élément qui n'a pas d' [span](#) attribut.

##### Modèle de contenu :

[Rien](#) (pour plus de précisions, [voir l'exemple](#) ).

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

[Exposed=Window]

```

interface HTMLTemplateElement : HTMLElement {

  [HTMLConstructor] constructor();

  readonly attribute DocumentFragment content;

};

```

L' template élément est utilisé pour déclarer des fragments de HTML qui peuvent être clonés et insérés dans le document par script.

Dans un rendu, l' template élément ne représente rien.

Le contenu du modèle d'un template élément ne sont pas des enfants de l'élément lui-même .

*Il est également possible, à la suite d'une manipulation du DOM, qu'un template élément contienne Text des nœuds et des nœuds d'élément ; cependant, en avoir est une violation du template modèle de contenu de l'élément, puisque son modèle de contenu est défini comme rien .*

Prenons par exemple le document suivant :

```

<!doctype html>
<html lang="en">
  <head>
    <title>Homework</title>
  <body>
    <template id="template"><p>Smile!</p></template>
    <script>
      let num = 3;
      const fragment =
document.getElementById('template').content.cloneNode(true);
      while (num-- > 1) {

fragment.firstChild.before(fragment.firstChild.cloneNode(true));
        fragment.firstChild.textContent +=
fragment.lastChild.textContent;
      }
      document.body.appendChild(fragment);
    </script>
  </html>

```

L'élément dans template n'est pas un enfant de template dans le DOM ; c'est un enfant de DocumentFragment renvoyé par l'attribut IDL templatede l'élément content.

Si le script devait appeler appendChild() l'élément template, cela ajouterait un enfant à l'élément template (comme pour tout autre élément) ; cependant, cela constitue une violation du modèle de contenu de l'élément.

**template.content**

✓

Renvoie le contenu du modèle (a DocumentFragment).

Chaque template élément a un DocumentFragment objet associé qui est son **contenu de modèle**. Le contenu du modèle n'a aucune exigence de conformité. Lorsqu'un template élément est créé, l'agent utilisateur doit exécuter les étapes suivantes pour établir le contenu du modèle :

1. Soit *doc* le document propriétaire du contenu du modèle approprié du nœud documenttemplate de l'élément.
2. Créez un DocumentFragment objet dont le nœud document est *doc* et host est l'élément template.
3. Définissez le contenu du modèletemplate de l'élément sur l'objet nouvellement créé. DocumentFragment

Le **document propriétaire du contenu du modèle approprié** d' Document *un doc* est renvoyé par l'algorithme suivant : Document

1. Si *doc* n'est pas Document créé par cet algorithme, alors :
  1. Si *doc* n'a pas encore de **modèle de document inerte associé**, alors :
    1. Soit *new doc* un new Document (dont le contexte de navigation est nul). Ceci est "un Document créé par cet algorithme" aux fins de l'étape ci-dessus.
    2. Si *doc* est un document HTML, marquez également le nouveau *doc* comme document HTML.
    3. Soit le document modèle inerte associé à *doc* comme nouveau *doc*.
  2. Définissez *doc* sur le modèle de document inerte associé à *doc*.

Chaque élément Document non créé par cet algorithme obtient ainsi un seul Document pour agir comme son proxy pour posséder le contenu du modèle de tous ses template éléments, afin qu'ils ne soient pas dans un contexte de navigation et restent donc inertes (par exemple, les scripts ne

*s'exécutent pas). Pendant ce temps, template les éléments à l'intérieur Document des objets créés par cet algorithme réutilisent simplement le même Document propriétaire pour leur contenu.*

## 2. Retour *doc* .

Les étapes d'adoption (avec *node* et *oldDocument* en paramètres) des template éléments sont les suivantes :

1. Soit *doc* le document propriétaire du contenu du modèle approprié du *nœud* du document .

*Le document node de *node* est l' Document objet dans lequel *node* vient d'être adopté .*

2. Adoptez le contenu du modèle de *node* (un objet) dans *doc* .DocumentFragment

L' **content** attribut IDL doit renvoyer le contenu du modèletemplate de l'élément .

---

Les étapes de clonage d'un *nœud*template d'élément en cours de clonage vers une copie *doivent* exécuter les étapes suivantes :

1. Si l' *indicateur clone children* n'est pas défini dans l' algorithme de clonage appelant , return.
2. Supposons que *le contenu copié* soit le résultat du clonage de tous les enfants du contenu du modèle de *nœud* , avec *document* défini sur le document de nœud du contenu du modèle de *copie* et avec l' *indicateur clone children* défini.
3. Ajouter *le contenu copié* au contenu du modèle de *copie* .

Dans cet exemple, un script remplit un tableau à quatre colonnes avec les données d'une structure de données, en utilisant un template pour fournir la structure de l'élément au lieu de générer manuellement la structure à partir du balisage.

```
<!DOCTYPE html>
<html lang='en'>
<title>Cat data</title>
<script>
  // Data is hard-coded here, but could come from the server
  var data = [
```

```

    { name: 'Pillar', color: 'Ticked Tabby', sex: 'Female
(neutered)', legs: 3 },
    { name: 'Hedral', color: 'Tuxedo', sex: 'Male (neutered)', legs:
4 },
  ];
</script>
<table>
  <thead>
    <tr>
      <th>Name <th>Color <th>Sex <th>Legs
    </tr>
  </thead>
  <tbody>
    <template id="row">
      <tr><td><td><td><td>
    </template>
  </tbody>
</table>
<script>
  var template = document.querySelector('#row');
  for (var i = 0; i < data.length; i += 1) {
    var cat = data[i];
    var clone = template.content.cloneNode(true);
    var cells = clone.querySelectorAll('td');
    cells[0].textContent = cat.name;
    cells[1].textContent = cat.color;
    cells[2].textContent = cat.sex;
    cells[3].textContent = cat.legs;
    template.parentNode.appendChild(clone);
  }
</script>

```

Cet exemple utilise `cloneNode()` sur le `template` contenu de ; il aurait pu utiliser de manière équivalente `document.importNode()`, qui fait la même chose. La seule différence entre ces deux API est lorsque le [document du nœud](#) est mis à jour : avec `cloneNode()` il est mis à jour lorsque les nœuds sont ajoutés avec `appendChild()`, avec `document.importNode()` il est mis à jour lorsque les nœuds sont clonés.

#### 4.12.3.1 Interaction des `template` éléments avec XSLT et XPath

*Cette section est non normative.*



This specification does not define how XSLT and XPath interact with the [template](#) element. However, in the absence of another specification actually defining this, here are some guidelines for implementers, which are intended to be consistent with other processing described in this specification:

- An XSLT processor based on an XML parser that acts [as described in this specification](#) needs to act as if [template](#) elements contain as descendants their [template contents](#) for the purposes of the transform.
- An XSLT processor that outputs a DOM needs to ensure that nodes that would go into a [template](#) element are instead placed into the element's [template contents](#).
- XPath evaluation using the XPath DOM API when applied to a [Document](#) parsed using the [HTML parser](#) or the [XML parser](#) described in this specification needs to ignore [template contents](#).

#### 4.12.4 The [slot](#) element



##### [Categories:](#)

[Flow content](#).  
[Phrasing content](#).

##### [Contexts in which this element can be used:](#)

Where [phrasing content](#) is expected.

##### [Content model:](#)

[Transparent](#)

##### [Tag omission in text/html:](#)

Neither tag is omissible.

##### [Content attributes:](#)

[Global attributes](#)  
[name](#) — Name of shadow tree slot

##### [Accessibility considerations:](#)

[For authors](#).  
[For implementers](#).

##### [DOM interface:](#)

[Exposed=Window]

```

interface HTMLSlotElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute DOMString name;

    sequence<Node> assignedNodes(optional
AssignedNodesOptions options = {});

    sequence<Element> assignedElements(optional
AssignedNodesOptions options = {});

    undefined assign((Element or Text)... nodes);

};

dictionary AssignedNodesOptions {

    boolean flatten = false;

};

```

The `slot` element defines a [slot](#). It is typically used in a [shadow tree](#). A `slot` element [represents](#) its [assigned nodes](#), if any, and its contents otherwise.

The `name` content attribute may contain any string value. It represents a [slot's name](#).

*The `name` attribute is used to [assign slots](#) to other elements: a `slot` element with a `name` attribute creates a named [slot](#) to which any element is [assigned](#) if that element has a `slot` attribute whose value matches that `name` attribute's value, and the `slot` element is a child of the [shadow tree](#) whose [root's host](#) has that corresponding `slot` attribute value.*

**`slot.name`**

✓MDN

Can be used to get and set `slot's name`.

**`slot.assignedNodes()`**

✓MDN

Returns `slot's assigned nodes`.

**`slot.assignedNodes({ flatten: true })`**

Returns *slot*'s [assigned nodes](#), if any, and *slot*'s children otherwise, and does the same for any [slot](#) elements encountered therein, recursively, until there are no [slot](#) elements left.

`slot.assignedElements()`

✓MDN

Returns *slot*'s [assigned nodes](#), limited to elements.

`slot.assignedElements({ flatten: true })`

Returns the same as [assignedNodes\({ flatten: true }\)](#), limited to elements.

`slot.assign(...nodes)`

Sets *slot*'s [manually assigned nodes](#) to the given *nodes*.

The **name** IDL attribute must [reflect](#) the content attribute of the same name.

The [slot](#) element has **manually assigned nodes**, which is an [ordered set](#) of [slottables](#) set by [assign\(\)](#). This set is initially empty.

*The [manually assigned nodes](#) set can be implemented using weak references to the [slottables](#), because this set is not directly accessible from script.*

The **assignedNodes(options)** method steps are:

1. If *options*["[flatten](#)"] is false, then return [this](#)'s [assigned nodes](#).
2. Return the result of [finding flattened slottables](#) with [this](#).

The **assignedElements(options)** method steps are:

1. If *options*["[flatten](#)"] is false, then return [this](#)'s [assigned nodes](#), filtered to contain only [Element](#) nodes.
2. Return the result of [finding flattened slottables](#) with [this](#), filtered to contain only [Element](#) nodes.

MDN

The **assign(...nodes)** method steps are:

1. [For each](#) *node* of [this](#)'s [manually assigned nodes](#), set *node*'s [manual slot assignment](#) to null.
2. Let *nodesSet* be a new [ordered set](#).
3. [For each](#) *node* of *nodes*:

1. If *node*'s [manual slot assignment](#) refers to a [slot](#), then remove *node* from that [slot](#)'s [manually assigned nodes](#).
2. Définissez [l'assignation manuelle de l'emplacement](#) du *nœud* sur [ceci](#) .
3. [Ajouter](#) un *nœud* à *nodesSet* .
4. Définissez [ces](#) nœuds [assignés manuellement](#) sur *nodesSet* .
5. Exécutez assign [slottables pour un arbre](#) pour [cette](#) racine .

#### 4.12.5 L' **canvas** élément



##### Catégories :

[Contenu du flux](#) .  
[Contenu de la phrase](#) .  
[Contenu intégré](#) .  
[Contenu palpable](#) .

##### Contextes dans lesquels cet élément peut être utilisé :

Où [le contenu intégré](#) est attendu.

##### Modèle de contenu :

[Transparent](#) , mais sans descendants [de contenu interactif](#) à l'exception des éléments, [img](#) des éléments avec [usemap](#) des attributs, [button](#) des éléments, [input](#) des éléments dont [type](#) l'attribut est à l'état [Checkbox](#) ou [Radio Button](#) , [input](#) des éléments qui sont [des boutons](#) et [select](#) des éléments avec un [multiple](#) attribut ou une [taille d'affichage](#) supérieure à 1.

##### Omission de balise dans text/html :

Aucune des deux balises n'est omise.

##### Attributs de contenu :

[Attributs globaux](#)  
[width](#) — Dimensions horizontales  
[height](#) — Dimension verticale

##### Considérations d'accessibilité :

[Pour les auteurs](#) .  
[Pour les exécutants](#) .

##### Interface DOM :

```

typedef (CanvasRenderingContext2D or
ImageBitmapRenderingContext or WebGLRenderingContext or
WebGL2RenderingContext or GPUCanvasContext)
RenderingContext;

[Exposed=Window]

interface HTMLCanvasElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute unsigned long width;

    [CEReactions] attribute unsigned long height;

    RenderingContext? getContext(DOMString contextId,
optional any options = null);

    USVString toDataURL(optional DOMString type =
"image/png", optional any quality);

    undefined toBlob(BlobCallback _callback, optional
DOMString type = "image/png", optional any quality);

    OffscreenCanvas transferControlToOffscreen();

};

callback BlobCallback = undefined (Blob? blob);

```

L' [canvas](#) élément fournit des scripts avec un canevas bitmap dépendant de la résolution, qui peut être utilisé pour le rendu de graphiques, de graphiques de jeu, d'art ou d'autres images visuelles à la volée.

Les auteurs ne doivent pas utiliser l' [canvas](#) élément dans un document lorsqu'un élément plus approprié est disponible. Par exemple, il est inapproprié d'utiliser un [canvas](#) élément pour rendre un en-tête de page : si la présentation souhaitée de l'en-tête est graphiquement intense, elle doit être balisée à l'aide d'éléments appropriés (généralement ), [h1](#) puis stylisée à l'aide de CSS et de technologies de support telles que [les arbres d'ombre](#) . .

Lorsque les auteurs utilisent l' [canvas](#) élément, ils doivent également fournir un contenu qui, lorsqu'il est présenté à l'utilisateur, transmet essentiellement la même fonction ou le même objectif que le [canvas](#) bitmap de . Ce contenu peut être placé comme contenu de l' [canvas](#) élément. Le contenu de l' [canvas](#) élément, le cas échéant, est le [contenu de secours](#) de l'élément .

---

In interactive visual media, if [scripting is enabled](#) for the [canvas](#) element, and if support for [canvas](#) elements has been enabled, then the [canvas](#) element [represents embedded content](#) consisting of a dynamically created image, the element's bitmap.

In non-interactive, static, visual media, if the [canvas](#) element has been previously associated with a rendering context (e.g. if the page was viewed in an interactive visual medium and is now being printed, or if some script that ran during the page layout process painted on the element), then the [canvas](#) element [represents embedded content](#) with the element's current bitmap and size. Otherwise, the element represents its [fallback content](#) instead.

In non-visual media, and in visual media if [scripting is disabled](#) for the [canvas](#) element or if support for [canvas](#) elements has been disabled, the [canvas](#) element [represents](#) its [fallback content](#) instead.

When a [canvas](#) element [represents embedded content](#), the user can still focus descendants of the [canvas](#) element (in the [fallback content](#)). When an element is [focused](#), it is the target of keyboard interaction events (even though the element itself is not visible). This allows authors to make an interactive canvas keyboard-accessible: authors should have a one-to-one mapping of interactive regions to [focusable areas](#) in the [fallback content](#). (Focus has no effect on mouse interaction events.) [\[UIEVENTS\]](#)

An element whose nearest [canvas](#) element ancestor is [being rendered](#) and [represents embedded content](#) is an element that is **being used as relevant canvas fallback content**.

---

The [canvas](#) element has two attributes to control the size of the element's bitmap: **width** and **height**. These attributes, when specified, must have values that are [valid non-negative integers](#). The [rules for parsing non-negative integers](#) must be used to **obtain their numeric values**. If an attribute is missing, or if parsing its value returns an error, then the default value must be used instead. The [width](#) attribute defaults to 300, and the [height](#) attribute defaults to 150.

When setting the value of the [width](#) or [height](#) attribute, if the [context mode](#) of the [canvas](#) element is set to [placeholder](#), the user agent must throw an ["InvalidStateError"](#) [DOMException](#) and leave the attribute's value unchanged.

The [intrinsic dimensions](#) of the [canvas](#) element when it [represents embedded content](#) are equal to the dimensions of the element's bitmap.

The user agent must use a square pixel density consisting of one pixel of image data per coordinate space unit for the bitmaps of a [canvas](#) and its rendering contexts.

*A [canvas](#) element can be sized arbitrarily by a style sheet, its bitmap is then subject to the ['object-fit'](#) CSS property.*

---

The bitmaps of [canvas](#) elements, the bitmaps of [ImageBitmap](#) objects, as well as some of the bitmaps of rendering contexts, such as those described in the sections on the [CanvasRenderingContext2D](#) and [ImageBitmapRenderingContext](#) objects below, have an **origin-clean** flag, which can be set to true or false. Initially, when the [canvas](#) element or [ImageBitmap](#) object is created, its bitmap's [origin-clean](#) flag must be set to true.

A [canvas](#) element can have a rendering context bound to it. Initially, it does not have a bound rendering context. To keep track of whether it has a rendering context or not, and what kind of rendering context it is, a [canvas](#) also has a **canvas context mode**, which is initially **none** but can be changed to either **placeholder**, **2d**, **bitmaprenderer**, **webgl**, **webgl2**, or **webgpu** by algorithms defined in this specification.

When its [canvas context mode](#) is **none**, a [canvas](#) element has no rendering context, and its bitmap must be [transparent black](#) with an [intrinsic width](#) equal to [the numeric value](#) of the element's [width](#) attribute and an [intrinsic height](#) equal to [the numeric value](#) of the element's [height](#) attribute, those values being interpreted in [CSS pixels](#), and being updated as the attributes are set, changed, or removed.

When its [canvas context mode](#) is [placeholder](#), a [canvas](#) element has no rendering context. It serves as a placeholder for an [OffscreenCanvas](#) object, and the content

of the [canvas](#) element is updated by calling the [commit\(\)](#) method of the [OffscreenCanvas](#) object's rendering context.

When a [canvas](#) element represents [embedded content](#), it provides a [paint source](#) whose width is the element's [intrinsic width](#), whose height is the element's [intrinsic height](#), and whose appearance is the element's bitmap.

Whenever the [width](#) and [height](#) content attributes are set, removed, changed, or redundantly set to the value they already have, then the user agent must perform the action from the row of the following table that corresponds to the [canvas](#) element's [context mode](#).

<a href="#">Context Mode</a>	Action
<a href="#">2d</a>	Follow the steps to <a href="#">set bitmap dimensions</a> to <a href="#">the numeric values</a> of the <a href="#">width</a> and <a href="#">height</a> content attributes.
<a href="#">webgl</a> or <a href="#">webgl2</a>	Follow the behavior defined in the WebGL specifications. <a href="#">[WEBGL]</a>
<a href="#">webgpu</a>	Follow the behavior defined in <i>WebGPU</i> . <a href="#">[WEBGPU]</a>
<a href="#">bitmaprenderer</a>	If the context's <a href="#">bitmap mode</a> is set to <a href="#">blank</a> , run the steps to <a href="#">set an ImageBitmapRenderingContext's output bitmap</a> , passing the <a href="#">canvas</a> element's rendering context.
<a href="#">placeholder</a>	Do nothing.
<a href="#">none</a>	Do nothing.



The [width](#) and [height](#) IDL attributes must [reflect](#) the respective content attributes of the same name, with the same defaults.

---



```
context = canvas.getContext(contextId [, options ])
```



Returns an object that exposes an API for drawing on the canvas. *contextId* specifies the desired API: "[2d](#)", "[bitmaprenderer](#)", "[webgl](#)", "[webgl2](#)", or "[webgpu](#)". *options* is handled by that API.

This specification defines the "[2d](#)" and "[bitmaprenderer](#)" contexts below. The WebGL specifications define the "[webgl](#)" and "[webgl2](#)" contexts. *WebGPU* defines the "[webgpu](#)" context. [\[WEBGL\]](#) [\[WEBGPU\]](#)

Returns null if *contextId* is not supported, or if the canvas has already been initialized with another context type (e.g., trying to get a "[2d](#)" context after getting a "[webgl](#)" context).

The `getContext(contextId, options)` method of the [canvas](#) element, when invoked, must run these steps:

1. If *options* is not an [object](#), then set *options* to null.
2. Set *options* to the result of [converting options](#) to a JavaScript value.
3. Run the steps in the cell of the following table whose column header matches this [canvas](#) element's [canvas context mode](#) and whose row header matches *contextId*:

	<a href="#">none</a>	<a href="#">2d</a>	<a href="#">bitma prend erer</a>	<a href="#">webgl or we bgl2</a>	<a href="#">we bg pu</a>	<a href="#">placeholder</a>
" <a href="#">2d</a> "	Follow the <a href="#">2D context creation algorithm</a> defined in the section below, passing it this <a href="#">canvas</a> element and <i>options</i> , to obtain a <a href="#">CanvasRenderingContext2D</a> object; if this does not throw an exception, then set this <a href="#">canvas</a> element's <a href="#">context mode</a> to <a href="#">2d</a> , and return the <a href="#">CanvasRenderingContext2D</a> object.	Ret urn the sa me obj ect as wa s ret urn ed the last tim e the met hod wa s	Retur n null.	Retur n null.	Ret urn nul l.	Throw an " <a href="#">InvalidSta teError</a> " <a href="#">DOME xception</a> .

	<a href="#">none</a>	<a href="#">2d</a>	<a href="#">bitmap renderer</a>	<a href="#">webgl or webgl2</a>	<a href="#">webgl renderer</a>	<a href="#">placeholder</a>
		invoked with this same first argument.				
" <a href="#">bitmap renderer</a> "	Follow the <a href="#">ImageBitmapRenderingContext creation algorithm</a> defined in the section below, passing it this <a href="#">canvas</a> element and <i>options</i> , to obtain an <a href="#">ImageBitmapRenderingContext</a> object; then set this <a href="#">canvas</a> element's <a href="#">context mode</a> to <a href="#">bitmaprenderer</a> , and return the <a href="#">ImageBitmapRenderingContext</a> object.	Return null.	Return the same object as was returned the last time the method was invoked with this same first argument.	Return null.	Return null.	Throw an " <a href="#">InvalidStateError</a> " <a href="#">DOMException</a> .
" <a href="#">webgl1</a> " or " <a href="#">webgl2</a> ", if the user agent supports the WebGL feature	Follow the instructions given in the WebGL specifications' <i>Context Creation</i> sections to obtain a <a href="#">WebGLRenderingContext</a> , <a href="#">WebGL2RenderingContext</a> , or null; if the returned value is null, then return null; otherwise, set this <a href="#">canvas</a> element's <a href="#">context mode</a> to <a href="#">webgl</a> or <a href="#">webgl2</a> , and return	Return null.	Return null.	Return the same object as was returned the last time the method	Return null.	Throw an " <a href="#">InvalidStateError</a> " <a href="#">DOMException</a> .

	<a href="#">none</a>	<a href="#">2d</a>	<a href="#">bitmap renderer</a>	<a href="#">webgl or webgl2</a>	<a href="#">webgpu</a>	<a href="#">placeholder</a>
in its current configuration	the <a href="#">WebGLRenderingContext</a> or <a href="#">WebGL2RenderingContext</a> object. <a href="#">[WEBGL]</a>			d was invoked with this same first argument.		
"webgpu", if the user agent supports the WebGPU feature in its current configuration	Follow the instructions given in <i>WebGPU</i> 's <a href="#">Canvas Rendering</a> section to obtain a <a href="#">GPUCanvasContext</a> or null; if the returned value is null, then return null; otherwise, set this <a href="#">canvas</a> element's <a href="#">context mode</a> to <a href="#">webgpu</a> and return the <a href="#">GPUCanvasContext</a> object. <a href="#">[WEBGPU]</a>	Return null.	Return null.	Return null.	Return the same object as was returned the last time the method was invoked with this same first argument.	Throw an <a href="#">"InvalidStateError"</a> <a href="#">DOMException</a> .
An unsupported	Return null.	Return null.	Return null.	Return null.	Return null.	Throw an <a href="#">"InvalidStateError"</a>

	<a href="#">none</a>	<a href="#">2d</a>	<a href="#">bitmap</a>	<a href="#">webgl</a>	<a href="#">webgl</a>	<a href="#">placeholder</a>
ported value*		null.			null.	<a href="#">TypeError</a> "DOMException".

4. \* For example, the "[webgl](#)" or "[webgl2](#)" value in the case of a user agent having exhausted the graphics hardware's abilities and having no software fallback implementation.

```
url = canvas.toDataURL([ type [, quality ] ])
```

✓MDN

Returns a [data: URL](#) for the image in the canvas.

The first argument, if provided, controls the type of the image to be returned (e.g. PNG or JPEG). The default is "[image/png](#)"; that type is also used if the given type isn't supported. The second argument applies if the type is an image format that supports variable quality (such as "[image/jpeg](#)"), and is a number in the range 0.0 to 1.0 inclusive indicating the desired quality level for the resulting image.

When trying to use types other than "[image/png](#)", authors can check if the image was really returned in the requested format by checking to see if the returned string starts with one of the exact strings "data:image/png," or "data:image/png;". If it does, the image is PNG, and thus the requested type was not supported. (The one exception to this is if the canvas has either no height or no width, in which case the result might simply be "data:,.")

```
canvas.toBlob(callback [, type [, quality ] ])
```

✓MDN

Creates a [Blob](#) object representing a file containing the image in the canvas, and invokes a callback with a handle to that object.

The second argument, if provided, controls the type of the image to be returned (e.g. PNG or JPEG). The default is "[image/png](#)"; that type is also used if the given type isn't supported. The third argument applies if the type is an image format that supports variable quality (such as "[image/jpeg](#)"), and is a number in the range 0.0 to 1.0 inclusive indicating the desired quality level for the resulting image.

```
canvas.transferControlToOffscreen()
```

MDN

Returns a newly created [OffscreenCanvas](#) object that uses the [canvas](#) element as a placeholder. Once the [canvas](#) element has become

a placeholder for an [OffscreenCanvas](#) object, its intrinsic size can no longer be changed, and it cannot have a rendering context. The content of the placeholder canvas is updated by calling the [commit\(\)](#) method of the [OffscreenCanvas](#) object's rendering context.

The [toDataURL\(type, quality\)](#) method, when invoked, must run these steps:

1. If this [canvas](#) element's bitmap's [origin-clean](#) flag is set to false, then throw a ["SecurityError" DOMException](#).
2. If this [canvas](#) element's bitmap has no pixels (i.e. either its horizontal dimension or its vertical dimension is zero) then return the string "data:,". (This is the shortest [data: URL](#); it represents the empty string in a `text/plain` resource.)
3. Let *file* be [a serialization of this canvas element's bitmap as a file](#), passing *type* and *quality* if given.
4. If *file* is null then return "data:,".
5. Return a [data: URL](#) representing *file*. [\[RFC2397\]](#)

The [toBlob\(callback, type, quality\)](#) method, when invoked, must run these steps:

1. If this [canvas](#) element's bitmap's [origin-clean](#) flag is set to false, then throw a ["SecurityError" DOMException](#).
2. Let *result* be null.
3. If this [canvas](#) element's bitmap has pixels (i.e., neither its horizontal dimension nor its vertical dimension is zero), then set *result* to a copy of this [canvas](#) element's bitmap.
4. Run these steps [in parallel](#):
  1. If *result* is non-null, then set *result* to [a serialization of result as a file](#) with *type* and *quality* if given.
  2. [Queue an element task](#) on the **canvas blob serialization task source** given the [canvas](#) element to run these steps:
    1. If *result* is non-null, then set *result* to a new [Blob](#) object, created in the [relevant realm](#) of this [canvas](#) element, representing *result*. [\[FILEAPI\]](#)
    2. [Invoke](#) *callback* with « *result* ».

The [transferControlToOffscreen\(\)](#) method, when invoked, must run these steps:

1. If this [canvas](#) element's [context mode](#) is not set to [none](#), throw an ["InvalidStateError"](#) [DOMException](#).
2. Let *offscreenCanvas* be a new [OffscreenCanvas](#) object with its width and height equal to the values of the [width](#) and [height](#) content attributes of this [canvas](#) element.
3. Set the [placeholder canvas element](#) of *offscreenCanvas* to a weak reference to this [canvas](#) element.
4. Set this [canvas](#) element's [context mode](#) to [placeholder](#).
5. Return *offscreenCanvas*.

#### 4.12.5.1 The 2D rendering context



```
typedef (HTMLImageElement or
```

```
SVGImageElement) HTMLOrSVGImageElement;
```

```
typedef (HTMLOrSVGImageElement or
```

```
HTMLVideoElement or
```

```
HTMLCanvasElement or
```

```
ImageBitmap or
```

```
OffscreenCanvas or
```

```
VideoFrame) CanvasImageSource;
```

```
enum PredefinedColorSpace { "srgb", "display-p3" };
```

```
enum CanvasFillRule { "nonzero", "evenodd" };
```

```
dictionary CanvasRenderingContext2DSettings {
```

```
    boolean alpha = true;
```

```
    boolean desynchronized = false;
```

```
    PredefinedColorSpace colorSpace = "srgb";
```

```
    boolean willReadFrequently = false;
```

```
};
```

```
enum ImageSmoothingQuality { "low", "medium", "high" };
```

```
[Exposed=Window]
```

```
interface CanvasRenderingContext2D {
```

```
    // back-reference to the canvas
```

```
    readonly attribute HTMLCanvasElement canvas;
```

```
    CanvasRenderingContext2DSettings getContextAttributes();
```

```
};
```

```
CanvasRenderingContext2D includes CanvasState;
```

```
CanvasRenderingContext2D includes CanvasTransform;
```

```
CanvasRenderingContext2D includes CanvasCompositing;
```

```
CanvasRenderingContext2D includes CanvasImageSmoothing;
```

```
CanvasRenderingContext2D includes CanvasFillStrokeStyles;
```

```
CanvasRenderingContext2D includes CanvasShadowStyles;
```

```
CanvasRenderingContext2D includes CanvasFilters;
```

```
CanvasRenderingContext2D includes CanvasRect;
```

[CanvasRenderingContext2D](#) includes [CanvasDrawPath](#);

[CanvasRenderingContext2D](#) includes [CanvasUserInterface](#);

[CanvasRenderingContext2D](#) includes [CanvasText](#);

[CanvasRenderingContext2D](#) includes [CanvasDrawImage](#);

[CanvasRenderingContext2D](#) includes [CanvasImageData](#);

[CanvasRenderingContext2D](#) includes [CanvasPathDrawingStyles](#);

[CanvasRenderingContext2D](#) includes [CanvasTextDrawingStyles](#);

[CanvasRenderingContext2D](#) includes [CanvasPath](#);

```
interface mixin CanvasState {
```

```
    // state
```

```
    undefined save(); // push state on state stack
```

```
    undefined restore(); // pop state stack and restore state
```

```
    undefined reset(); // reset the rendering context to its
```

```
    default state
```

```
    boolean isContextLost(); // return whether context is lost
```

```
};
```

```
interface mixin CanvasTransform {
```

```
    // transformations (default transform is the identity matrix)
```

```
    undefined scale(unrestricted double x, unrestricted double y);
```

```
    undefined rotate(unrestricted double angle);
```

```
    undefined translate(unrestricted double x, unrestricted double
```

```
    y);
```



```
undefined transform(unrestricted double a, unrestricted double  
b, unrestricted double c, unrestricted double d, unrestricted  
double e, unrestricted double f);
```

```
[NewObject] DOMMatrix getTransform();
```

```
undefined setTransform(unrestricted double a, unrestricted  
double b, unrestricted double c, unrestricted double d,  
unrestricted double e, unrestricted double f);
```

```
undefined setTransform(optional DOMMatrix2DInit transform =  
{});
```

```
undefined resetTransform();
```

```
};
```

```
interface mixin CanvasCompositing {
```

```
    // compositing
```

```
    attribute unrestricted double globalAlpha; // (default 1.0)
```

```
    attribute DOMString globalCompositeOperation; // (default  
"source-over")
```

```
};
```

```
interface mixin CanvasImageSmoothing {
```

```
    // image smoothing
```

```
    attribute boolean imageSmoothingEnabled; // (default true)
```

```
    attribute ImageSmoothingQuality imageSmoothingQuality; //  
(default low)
```

```
};
```

```
interface mixin CanvasFillStrokeStyles {
```

```
    // colors and styles (see also the CanvasPathDrawingStyles and  
    CanvasTextDrawingStyles interfaces)
```

```
    attribute (DOMString or CanvasGradient or CanvasPattern)  
    strokeStyle; // (default black)
```

```
    attribute (DOMString or CanvasGradient or CanvasPattern)  
    fillStyle; // (default black)
```

```
    CanvasGradient createLinearGradient(double x0, double y0,  
double x1, double y1);
```

```
    CanvasGradient createRadialGradient(double x0, double y0,  
double r0, double x1, double y1, double r1);
```

```
    CanvasGradient createConicGradient(double startAngle, double x,  
double y);
```

```
    CanvasPattern? createPattern(CanvasImageSource image,  
[LegacyNullToEmptyString] DOMString repetition);
```

```
};
```

```
interface mixin CanvasShadowStyles {
```

```
    // shadows
```

```
    attribute unrestricted double shadowOffsetX; // (default 0)
```

```
    attribute unrestricted double shadowOffsetY; // (default 0)
```

```
    attribute unrestricted double shadowBlur; // (default 0)
```

```
attribute DOMString shadowColor; // (default transparent black)
```

```
};
```

```
interface mixin CanvasFilters {
```

```
    // filters
```

```
    attribute DOMString filter; // (default "none")
```

```
};
```

```
interface mixin CanvasRect {
```

```
    // rects
```

```
    undefined clearRect(unrestricted double x, unrestricted double  
y, unrestricted double w, unrestricted double h);
```

```
    undefined fillRect(unrestricted double x, unrestricted double  
y, unrestricted double w, unrestricted double h);
```

```
    undefined strokeRect(unrestricted double x, unrestricted double  
y, unrestricted double w, unrestricted double h);
```

```
};
```

```
interface mixin CanvasDrawPath {
```

```
    // path API (see also CanvasPath)
```

```
    undefined beginPath();
```

```
    undefined fill(optional CanvasFillRule fillRule = "nonzero");
```

```
    undefined fill(Path2D path, optional CanvasFillRule fillRule =  
"nonzero");
```

```
undefined stroke();
```

```
undefined stroke(Path2D path);
```

```
undefined clip(optional CanvasFillRule fillRule = "nonzero");
```

```
undefined clip(Path2D path, optional CanvasFillRule fillRule =  
"nonzero");
```

```
boolean isPointInPath(unrestricted double x, unrestricted  
double y, optional CanvasFillRule fillRule = "nonzero");
```

```
boolean isPointInPath(Path2D path, unrestricted double x,  
unrestricted double y, optional CanvasFillRule fillRule =  
"nonzero");
```

```
boolean isPointInStroke(unrestricted double x, unrestricted  
double y);
```

```
boolean isPointInStroke(Path2D path, unrestricted double x,  
unrestricted double y);
```

```
};
```

```
interface mixin CanvasUserInterface {
```

```
undefined drawFocusIfNeeded(Element element);
```

```
undefined drawFocusIfNeeded(Path2D path, Element element);
```

```
undefined scrollPathIntoView();
```

```
undefined scrollPathIntoView(Path2D path);
```

```
};
```

```
interface mixin CanvasText {
```

```
// text (see also the CanvasPathDrawingStyles and  
CanvasTextDrawingStyles interfaces)
```

```
undefined fillText(DOMString text, unrestricted double x,  
unrestricted double y, optional unrestricted double maxWidth);
```

```
undefined strokeText(DOMString text, unrestricted double x,  
unrestricted double y, optional unrestricted double maxWidth);
```

```
TextMetrics measureText(DOMString text);
```

```
};
```

```
interface mixin CanvasDrawImage {
```

```
    // drawing images
```

```
    undefined drawImage(CanvasImageSource image, unrestricted  
double dx, unrestricted double dy);
```

```
    undefined drawImage(CanvasImageSource image, unrestricted  
double dx, unrestricted double dy, unrestricted double dw,  
unrestricted double dh);
```

```
    undefined drawImage(CanvasImageSource image, unrestricted  
double sx, unrestricted double sy, unrestricted double sw,  
unrestricted double sh, unrestricted double dx, unrestricted  
double dy, unrestricted double dw, unrestricted double dh);
```

```
};
```

```
interface mixin CanvasImageData {
```

```
    // pixel manipulation
```

```
ImageData createImageData([EnforceRange] long sw,  
[EnforceRange] long sh, optional ImageDataSettings settings =  
{});
```

```
ImageData createImageData(ImageData imagedata);
```

```
ImageData getImageData([EnforceRange] long sx, [EnforceRange]  
long sy, [EnforceRange] long sw, [EnforceRange] long sh, optional  
ImageDataSettings settings = {});
```

```
undefined putImageData(ImageData imagedata, [EnforceRange] long  
dx, [EnforceRange] long dy);
```

```
undefined putImageData(ImageData imagedata, [EnforceRange] long  
dx, [EnforceRange] long dy, [EnforceRange] long dirtyX,  
[EnforceRange] long dirtyY, [EnforceRange] long dirtyWidth,  
[EnforceRange] long dirtyHeight);
```

```
};
```

```
enum CanvasLineCap { "butt", "round", "square" };
```

```
enum CanvasLineJoin { "round", "bevel", "miter" };
```

```
enum CanvasTextAlign { "start", "end", "left", "right", "center"  
};
```

```
enum CanvasTextBaseline { "top", "hanging", "middle",  
"alphabetic", "ideographic", "bottom" };
```

```
enum CanvasDirection { "ltr", "rtl", "inherit" };
```

```
enum CanvasFontKerning { "auto", "normal", "none" };
```

```
enum CanvasFontStretch { "ultra-condensed", "extra-condensed",  
"condensed", "semi-condensed", "normal", "semi-expanded",  
"expanded", "extra-expanded", "ultra-expanded" };
```

```
enum CanvasFontVariantCaps { "normal", "small-caps", "all-small-  
caps", "petite-caps", "all-petite-caps", "unicase", "titling-  
caps" };
```

```
enum CanvasTextRendering { "auto", "optimizeSpeed",  
"optimizeLegibility", "geometricPrecision" };
```

```
interface mixin CanvasPathDrawingStyles {
```

```
    // line caps/joins
```

```
    attribute unrestricted double lineWidth; // (default 1)
```

```
    attribute CanvasLineCap lineCap; // (default "butt")
```

```
    attribute CanvasLineJoin lineJoin; // (default "miter")
```

```
    attribute unrestricted double miterLimit; // (default 10)
```

```
    // dashed lines
```

```
    undefined setLineDash(sequence<unrestricted double> segments);
```

```
    // default empty
```

```
    sequence<unrestricted double> getLineDash();
```

```
    attribute unrestricted double lineDashOffset;
```

```
};
```

```
interface mixin CanvasTextDrawingStyles {
```

```
    // text
```

```
attribute DOMString font; // (default 10px sans-serif)
```

```
attribute CanvasTextAlign textAlign; // (default: "start")
```

```
attribute CanvasTextBaseline textBaseline; // (default:
```

```
"alphabetic")
```

```
attribute CanvasDirection direction; // (default: "inherit")
```

```
attribute DOMString letterSpacing; // (default: "0px")
```

```
attribute CanvasFontKerning fontKerning; // (default: "auto")
```

```
attribute CanvasFontStretch fontStretch; // (default: "normal")
```

```
attribute CanvasFontVariantCaps fontVariantCaps; // (default:
```

```
"normal")
```

```
attribute CanvasTextRendering textRendering; // (default:
```

```
"auto")
```

```
attribute DOMString wordSpacing; // (default: "0px")
```

```
};
```

```
interface mixin CanvasPath {
```

```
    // shared path API methods
```

```
    undefined closePath();
```

```
    undefined moveTo(unrestricted double x, unrestricted double y);
```

```
    undefined lineTo(unrestricted double x, unrestricted double y);
```

```
    undefined quadraticCurveTo(unrestricted double cpx,
```

```
    unrestricted double cpy, unrestricted double x, unrestricted
```

```
    double y);
```



```
undefined bezierCurveTo(unrestricted double cplx, unrestricted  
double cply, unrestricted double cp2x, unrestricted double cp2y,  
unrestricted double x, unrestricted double y);
```

```
undefined arcTo(unrestricted double x1, unrestricted double y1,  
unrestricted double x2, unrestricted double y2, unrestricted  
double radius);
```

```
undefined rect(unrestricted double x, unrestricted double y,  
unrestricted double w, unrestricted double h);
```

```
undefined roundRect(unrestricted double x, unrestricted double  
y, unrestricted double w, unrestricted double h, optional  
(unrestricted double or DOMPointInit or sequence<(unrestricted  
double or DOMPointInit)>) radii = 0);
```

```
undefined arc(unrestricted double x, unrestricted double y,  
unrestricted double radius, unrestricted double startAngle,  
unrestricted double endAngle, optional boolean counterclockwise =  
false);
```

```
undefined ellipse(unrestricted double x, unrestricted double y,  
unrestricted double radiusX, unrestricted double radiusY,  
unrestricted double rotation, unrestricted double startAngle,  
unrestricted double endAngle, optional boolean counterclockwise =  
false);
```

```
};
```

```
[Exposed=(Window,Worker)]
```

```
interface CanvasGradient {
```

```
    // opaque object
```

```
undefined addColorStop(double offset, DOMString color);
```

```
};
```

```
[Exposed=(Window,Worker)]
```

```
interface CanvasPattern {
```

```
// opaque object
```

```
undefined setTransform(optional DOMMatrix2DInit transform =
```

```
{}));
```

```
};
```

```
[Exposed=(Window,Worker)]
```

```
interface TextMetrics {
```

```
// x-direction
```

```
readonly attribute double width; // advance width
```

```
readonly attribute double actualBoundingBoxLeft;
```

```
readonly attribute double actualBoundingBoxRight;
```

```
// y-direction
```

```
readonly attribute double fontBoundingBoxAscent;
```

```
readonly attribute double fontBoundingBoxDescent;
```

```
readonly attribute double actualBoundingBoxAscent;
```

```
readonly attribute double actualBoundingBoxDescent;
```

```
readonly attribute double emHeightAscent;
```

```
readonly attribute double emHeightDescent;
```

```
readonly attribute double hangingBaseline;
```

```
readonly attribute double alphabeticBaseline;
```

```
readonly attribute double ideographicBaseline;
```

```
};
```

```
dictionary ImageDataSettings {
```

```
  PredefinedColorSpace colorSpace;
```

```
};
```

```
[Exposed=(Window,Worker),
```

```
  Serializable]
```

```
interface ImageData {
```

```
  constructor(unsigned long sw, unsigned long sh, optional
```

```
  ImageDataSettings settings = {});
```

```
  constructor(Uint8ClampedArray data, unsigned long sw, optional
```

```
  unsigned long sh, optional ImageDataSettings settings = {});
```

```
  readonly attribute unsigned long width;
```

```
  readonly attribute unsigned long height;
```

```
  readonly attribute Uint8ClampedArray data;
```

```
  readonly attribute PredefinedColorSpace colorSpace;
```

```
};
```

```
[Exposed=(Window,Worker)]
```

```
interface Path2D {
```

```
constructor(optional (Path2D or DOMString) path);
```

```
undefined addPath(Path2D path, optional DOMMatrix2DInit
```

```
transform = {});
```

```
};
```

```
Path2D includes CanvasPath;
```

*To maintain compatibility with existing web content, user agents need to enumerate methods defined in [CanvasUserInterface](#) immediately after the [stroke\(\)](#) method on [CanvasRenderingContext2D](#) objects.*

```
context = canvas.getContext('2d' [, { [ alpha: true ]  
[, desynchronized: false ] [, colorSpace: 'srgb']  
[, willReadFrequently: false ] } ])
```

Returns a [CanvasRenderingContext2D](#) object that is permanently bound to a particular [canvas](#) element.

If the [alpha](#) member is false, then the context is forced to always be opaque.

If the [desynchronized](#) member is true, then the context might be [desynchronized](#).

The [colorSpace](#) member specifies the [color space](#) of the rendering context.

If the [willReadFrequently](#) member is true, then the context is marked for [readback optimization](#).

```
context.canvas
```



Returns the [canvas](#) element.

```
attributes = context.getContextAttributes()
```

Returns an object whose:

- [alpha](#) member is true if the context has an alpha channel, or false if it was forced to be opaque.
- [desynchronized](#) member is true if the context can be [desynchronized](#).
- [colorSpace](#) member is a string indicating the context's [color space](#).
- [willReadFrequently](#) member is true if the context is marked for [readback optimization](#).

A [CanvasRenderingContext2D](#) object has an **output bitmap** that is initialized when the object is created.

The [output bitmap](#) has an [origin-clean](#) flag, which can be set to true or false. Initially, when one of these bitmaps is created, its [origin-clean](#) flag must be set to true.

The [CanvasRenderingContext2D](#) object also has an **alpha** boolean. When a [CanvasRenderingContext2D](#) object's [alpha](#) is false, then its alpha channel must be fixed to 1.0 (fully opaque) for all pixels, and attempts to change the alpha component of any pixel must be silently ignored.

*Thus, the bitmap of such a context starts off as [opaque black](#) instead of [transparent black](#); [clearRect\(\)](#) always results in [opaque black](#) pixels, every fourth byte from [getImageData\(\)](#) is always 255, the [putImageData\(\)](#) method effectively ignores every fourth byte in its input, and so on. However, the alpha component of styles and images drawn onto the canvas are still honoured up to the point where they would impact the [output bitmap](#)'s alpha channel; for instance, drawing a 50% transparent white square on a freshly created [output bitmap](#) with its [alpha](#) set to false will result in a fully-opaque green square.*

The [CanvasRenderingContext2D](#) object also has a **desynchronized** boolean. When a [CanvasRenderingContext2D](#) object's [desynchronized](#) is true, then the user agent may optimize the rendering of the canvas to reduce the latency, as measured from input events to rasterization, by desynchronizing the canvas paint cycle from the event loop, bypassing the ordinary user agent rendering algorithm, or both. Insofar as this mode involves bypassing the usual paint mechanisms, rasterization, or both, it might introduce visible tearing artifacts.

*The user agent usually renders on a buffer which is not being displayed, quickly swapping it and the one being scanned out for presentation; the former buffer is called back buffer and the latter front buffer. A popular technique for reducing latency is called front buffer rendering, also known as single buffer rendering, where rendering happens in parallel and rapidly with the scanning out process. This technique reduces the latency at the price of potentially introducing tearing artifacts and can be used to implement in total or part of the [desynchronized](#) boolean. [\[MULTIPLEBUFFERING\]](#) The [desynchronized](#) boolean can be useful when implementing certain kinds of applications, such as drawing applications, where the latency between input and rasterization is critical.*

The [CanvasRenderingContext2D](#) object also has a **will read frequently** boolean. When a [CanvasRenderingContext2D](#) object's [will read frequently](#) is true, the user agent may optimize the canvas for readback operations.

*On most devices the user agent needs to decide whether to store the canvas's [output bitmap](#) on the GPU (this is also called "hardware accelerated"), or on the CPU (also called "software"). Most rendering operations are more performant for accelerated canvases, with the major exception being readback with [getImageData\(\)](#), [toDataURL\(\)](#), or [toBlob\(\)](#). [CanvasRenderingContext2D](#) objects with [will read frequently](#) equal to true tell the user agent that the webpage is likely to perform many readback operations and that it is advantageous to use a software canvas.*

The [CanvasRenderingContext2D](#) object also has a **color space** setting of type [PredefinedColorSpace](#). The [CanvasRenderingContext2D](#) object's [color space](#) indicates the color space for the [output bitmap](#).

The [getContextAttributes\(\)](#) method steps are to return «[ "[alpha](#)" → [this's alpha](#), "[desynchronized](#)" → [this's desynchronized](#), "[colorSpace](#)" → [this's color space](#), "[willReadFrequently](#)" → [this's will read frequently](#) ]».

---

The [CanvasRenderingContext2D](#) 2D rendering context represents a flat linear Cartesian surface whose origin (0,0) is at the top left corner, with the coordinate space having x values increasing when going right, and y values increasing when going down. The x-coordinate of the right-most edge is equal to the width of the rendering context's [output bitmap](#) in [CSS pixels](#); similarly, the y-coordinate of the bottom-most edge is equal to the height of the rendering context's [output bitmap](#) in [CSS pixels](#).

The size of the coordinate space does not necessarily represent the size of the actual bitmaps that the user agent will use internally or during rendering. On high-definition displays, for instance, the user agent may internally use bitmaps with four device pixels per unit in the coordinate space, so that the rendering remains at high quality throughout. Anti-aliasing can similarly be implemented using oversampling with bitmaps of a higher resolution than the final image on the display.

Using [CSS pixels](#) to describe the size of a rendering context's [output bitmap](#) does not mean that when rendered the canvas will cover an equivalent area in [CSS pixels](#). [CSS pixels](#) are reused for ease of integration with CSS features, such as text layout.

In other words, the [canvas](#) element below's rendering context has a 200x200 [output bitmap](#) (which internally uses [CSS pixels](#) as a unit for ease of integration with CSS) and is rendered as 100x100 [CSS pixels](#):

```
<canvas width=200 height=200 style=width:100px;height:100px>
```

---

The **2D context creation algorithm**, which is passed a *target* (a [canvas](#) element) and *options*, consists of running these steps:

1. Let *settings* be the result of [converting options](#) to the dictionary type [CanvasRenderingContext2DSettings](#). (This can throw an exception.).
  2. Let *context* be a new [CanvasRenderingContext2D](#) object.
  3. Initialize *context*'s [canvas](#) attribute to point to *target*.
  4. Set *context*'s [output bitmap](#) to the same bitmap as *target*'s bitmap (so that they are shared).
  5. [Set bitmap dimensions](#) to [the numeric values](#) of *target*'s [width](#) and [height](#) content attributes.
  6. Set *context*'s [alpha](#) to *settings*["[alpha](#)"].
  7. Set *context*'s [desynchronized](#) to *settings*["[desynchronized](#)"].
  8. Set *context*'s [color space](#) to *settings*["[colorSpace](#)"].
  9. Set *context*'s [will read frequently](#) to *settings*["[willReadFrequently](#)"].
  10. Return *context*.
- 

When the user agent is to **set bitmap dimensions** to *width* and *height*, it must run these steps:

1. [Reset the rendering context to its default state](#).
2. Resize the [output bitmap](#) to the new *width* and *height*.
3. Let *canvas* be the [canvas](#) element to which the rendering context's [canvas](#) attribute was initialized.
4. If [the numeric value](#) of *canvas*'s [width](#) content attribute differs from *width*, then set *canvas*'s [width](#) content attribute to the shortest possible string representing *width* as a [valid non-negative integer](#).
5. If [the numeric value](#) of *canvas*'s [height](#) content attribute differs from *height*, then set *canvas*'s [height](#) content attribute to the shortest possible string representing *height* as a [valid non-negative integer](#).

Only one square appears to be drawn in the following example:

```
// canvas is a reference to a <canvas> element
var context = canvas.getContext('2d');
```

```
context.fillRect(0,0,50,50);  
canvas.setAttribute('width', '300'); // clears the canvas  
context.fillRect(0,100,50,50);  
canvas.width = canvas.width; // clears the canvas  
context.fillRect(100,0,50,50); // only this square remains
```

---

The **canvas** attribute must return the value it was initialized to when the object was created.

---

The [PredefinedColorSpace](#) enumeration is used to specify the [color space](#) of the canvas's backing store.

The "**srgb**" value indicates the '[srgb](#)' color space.

The "**display-p3**" value indicates the '[display-p3](#)' color space.

*Algorithms for converting between color spaces are found in the [Predefined color spaces](#) section of CSS Color. [\[CSSCOLOR\]](#)*

---

The [CanvasFillRule](#) enumeration is used to select the **fill rule** algorithm by which to determine if a point is inside or outside a path.

The value "**nonzero**" value indicates the nonzero winding rule, wherein a point is considered to be outside a shape if the number of times a half-infinite straight line drawn from that point crosses the shape's path going in one direction is equal to the number of times it crosses the path going in the other direction.

The "**evenodd**" value indicates the even-odd rule, wherein a point is considered to be outside a shape if the number of times a half-infinite straight line drawn from that point crosses the shape's path is even.

If a point is not outside a shape, it is inside the shape.



---

The [ImageSmoothingQuality](#) enumeration is used to express a preference for the interpolation quality to use when smoothing images.

The "[low](#)" value indicates a preference for a low level of image interpolation quality. Low-quality image interpolation may be more computationally efficient than higher settings.

The "[medium](#)" value indicates a preference for a medium level of image interpolation quality.

The "[high](#)" value indicates a preference for a high level of image interpolation quality. High-quality image interpolation may be more computationally expensive than lower settings.

*Bilinear scaling is an example of a relatively fast, lower-quality image-smoothing algorithm. Bicubic or Lanczos scaling are examples of image-smoothing algorithms that produce higher-quality output. This specification does not mandate that specific interpolation algorithms be used.*

#### **4.12.5.1.1 Implementation notes**

*This section is non-normative.*

The [output bitmap](#), when it is not directly displayed by the user agent, implementations can, instead of updating this bitmap, merely remember the sequence of drawing operations that have been applied to it until such time as the bitmap's actual data is needed (for example because of a call to [drawImage\(\)](#), or the [createImageBitmap\(\)](#) factory method). In many cases, this will be more memory efficient.

The bitmap of a [canvas](#) element is the one bitmap that's pretty much always going to be needed in practice. The [output bitmap](#) of a rendering context, when it has one, is always just an alias to a [canvas](#) element's bitmap.

Additional bitmaps are sometimes needed, e.g. to enable fast drawing when the canvas is being painted at a different size than its [intrinsic size](#), or to enable double buffering so that graphics updates, like page scrolling for example, can be processed concurrently while canvas draw commands are being executed.

#### 4.12.5.1.2 The canvas state

Objects that implement the [CanvasState](#) interface maintain a stack of drawing states. **Drawing states** consist of:

- The current [transformation matrix](#).
- The current [clipping region](#).
- The current [letter spacing](#), [word spacing](#), [fill style](#), [stroke style](#), [filter](#), [global alpha](#), and [compositing and blending operator](#).
- The current values of the following attributes: [lineWidth](#), [lineCap](#), [lineJoin](#), [miterLimit](#), [lineDashOffset](#), [shadowOffsetX](#), [shadowOffsetY](#), [shadowBlur](#), [shadowColor](#), [font](#), [textAlign](#), [textBaseline](#), [direction](#), [fontKerning](#), [fontStretch](#), [fontVariantCaps](#), [textRendering](#), [imageSmoothingEnabled](#), [imageSmoothingQuality](#).
- The current [dash list](#).

*The rendering context's bitmaps are not part of the drawing state, as they depend on whether and how the rendering context is bound to a [canvas](#) element.*

Objects that implement the [CanvasState](#) mixin have a **context lost** boolean, that is initialized to false when the object is created. The [context lost](#) value is updated in the [context lost steps](#).

`context.save()`

✓MDN

Pushes the current state onto the stack.

`context.restore()`

✓MDN

Pops the top state on the stack, restoring the context to that state.

`context.reset()`

Resets the rendering context, which includes the backing buffer, the drawing state stack, path, and styles.

`context.isContextLost()`

Returns true if the rendering context was lost. Context loss can occur due to driver crashes, running out of memory, etc. In these cases, the canvas loses its backing storage and takes steps to [reset the rendering context to its default state](#).

The [save\(\)](#) method steps are to push a copy of the current drawing state onto the drawing state stack.

The [restore\(\)](#) method steps are to pop the top entry in the drawing state stack, and reset the drawing state it describes. If there is no saved state, then the method must do nothing.



The `reset()` method steps are to [reset the rendering context to its default state](#).

To **reset the rendering context to its default state**:

1. Clear canvas's bitmap to [transparent black](#).
2. Empty the list of subpaths in context's [current default path](#).
3. Clear the context's drawing state stack.
4. Reset everything that [drawing state](#) consists of to their initial values.



The `isContextLost()` method steps are to return [this](#)'s [context lost](#).

#### 4.12.5.1.3 Line styles

```
context.lineWidth [ = value ]
```



```
styles.lineWidth [ = value ]
```

Returns the current line width.

Can be set, to change the line width. Values that are not finite values greater than zero are ignored.

```
context.lineCap [ = value ]
```



```
styles.lineCap [ = value ]
```

Returns the current line cap style.

Can be set, to change the line cap style.

The possible line cap styles are "butt", "round", and "square". Other values are ignored.

```
context.lineJoin [ = value ]
```



```
styles.lineJoin [ = value ]
```

Returns the current line join style.

Can be set, to change the line join style.

The possible line join styles are "bevel", "round", and "miter". Other values are ignored.

```
context.miterLimit [ = value ]
```

✓MDN

```
styles.miterLimit [ = value ]
```

Returns the current miter limit ratio.

Can be set, to change the miter limit ratio. Values that are not finite values greater than zero are ignored.

```
context.setLineDash(segments)
```

✓MDN

```
styles.setLineDash(segments)
```

Sets the current line dash pattern (as used when stroking). The argument is a list of distances for which to alternately have the line on and the line off.

```
segments = context.getLineDash()
```

✓MDN

```
segments = styles.getLineDash()
```

Returns a copy of the current line dash pattern. The array returned will always have an even number of entries (i.e. the pattern is normalized).

```
context.lineDashOffset
```

✓MDN

```
styles.lineDashOffset
```

Returns the phase offset (in the same units as the line dash pattern).

Can be set, to change the phase offset. Values that are not finite values are ignored.

Objects that implement the [CanvasPathDrawingStyles](#) interface have attributes and methods (defined in this section) that control how lines are treated by the object.

The **lineWidth** attribute gives the width of lines, in coordinate space units. On getting, it must return the current value. On setting, zero, negative, infinite, and NaN values must be ignored, leaving the value unchanged; other values must change the current value to the new value.

When the object implementing the [CanvasPathDrawingStyles](#) interface is created, the **lineWidth** attribute must initially have the value 1.0.

---

The **lineCap** attribute defines the type of endings that UAs will place on the end of lines. The three valid values are "butt", "round", and "square".

On getting, it must return the current value. On setting, the current value must be changed to the new value.

When the object implementing the [CanvasPathDrawingStyles](#) interface is created, the [lineCap](#) attribute must initially have the value "butt".

---

The [lineJoin](#) attribute defines the type of corners that UAs will place where two lines meet. The three valid values are "bevel", "round", and "miter".

On getting, it must return the current value. On setting, the current value must be changed to the new value.

When the object implementing the [CanvasPathDrawingStyles](#) interface is created, the [lineJoin](#) attribute must initially have the value "miter".

---

When the [lineJoin](#) attribute has the value "miter", strokes use the miter limit ratio to decide how to render joins. The miter limit ratio can be explicitly set using the [miterLimit](#) attribute. On getting, it must return the current value. On setting, zero, negative, infinite, and NaN values must be ignored, leaving the value unchanged; other values must change the current value to the new value.

When the object implementing the [CanvasPathDrawingStyles](#) interface is created, the [miterLimit](#) attribute must initially have the value 10.0.

---

Each [CanvasPathDrawingStyles](#) object has a **dash list**, which is either empty or consists of an even number of non-negative numbers. Initially, the [dash list](#) must be empty.

The [setLineDash\(\*segments\*\)](#) method, when invoked, must run these steps:

1. If any value in *segments* is not finite (e.g. an Infinity or a NaN value), or if any value is negative (less than zero), then return (without throwing an exception;

user agents could show a message on a developer console, though, as that would be helpful for debugging).

2. If the number of elements in *segments* is odd, then let *segments* be the concatenation of two copies of *segments*.
3. Let the object's [dash list](#) be *segments*.

When the `getLineDash()` method is invoked, it must return a sequence whose values are the values of the object's [dash list](#), in the same order.

It is sometimes useful to change the "phase" of the dash pattern, e.g. to achieve a "marching ants" effect. The phase can be set using the `lineDashOffset` attribute. On getting, it must return the current value. On setting, infinite and NaN values must be ignored, leaving the value unchanged; other values must change the current value to the new value.

When the object implementing the [CanvasPathDrawingStyles](#) interface is created, the `lineDashOffset` attribute must initially have the value 0.0.

---

When a user agent is to **trace a path**, given an object *style* that implements the [CanvasPathDrawingStyles](#) interface, it must run the following algorithm. This algorithm returns a new [path](#).

1. Let *path* be a copy of the path being traced.
2. Prune all zero-length [line segments](#) from *path*.
3. Remove from *path* any subpaths containing no lines (i.e. subpaths with just one point).
4. Replace each point in each subpath of *path* other than the first point and the last point of each subpath by a *join* that joins the line leading to that point to the line leading out of that point, such that the subpaths all consist of two points (a starting point with a line leading out of it, and an ending point with a line leading into it), one or more lines (connecting the points and the joins), and zero or more joins (each connecting one line to another), connected together such that each subpath is a series of one or more lines with a join between each one and a point on each end.
5. Add a straight closing line to each closed subpath in *path* connecting the last point and the first point of that subpath; change the last point to a join (from the previously last line to the newly added closing line), and change the first point to a join (from the newly added closing line to the first line).

6. If *style*'s [dash list](#) is empty, then jump to the step labeled *convert*.
7. Let *pattern width* be the concatenation of all the entries of *style*'s [dash list](#), in coordinate space units.
8. For each subpath *subpath* in *path*, run the following substeps. These substeps mutate the subpaths in *path in vivo*.
  1. Let *subpath width* be the length of all the lines of *subpath*, in coordinate space units.
  2. Let *offset* be the value of *style*'s [lineDashOffset](#), in coordinate space units.
  3. While *offset* is greater than *pattern width*, decrement it by *pattern width*.  
While *offset* is less than zero, increment it by *pattern width*.
  4. Define *L* to be a linear coordinate line defined along all lines in *subpath*, such that the start of the first line in the subpath is defined as coordinate 0, and the end of the last line in the subpath is defined as coordinate *subpath width*.
  5. Let *position* be zero minus *offset*.
  6. Let *index* be 0.
  7. Let *current state* be *off* (the other states being *on* and *zero-on*).
  8. *Dash on*: Let *segment length* be the value of *style*'s [dash list](#)'s *index*th entry.
  9. Increment *position* by *segment length*.
  10. If *position* is greater than *subpath width*, then end these substeps for this subpath and start them again for the next subpath; if there are no more subpaths, then jump to the step labeled *convert* instead.
  11. If *segment length* is nonzero, then let *current state* be *on*.
  12. Increment *index* by one.
  13. *Dash off*: Let *segment length* be the value of *style*'s [dash list](#)'s *index*th entry.
  14. Let *start* be the offset *position* on *L*.
  15. Increment *position* by *segment length*.
  16. If *position* is less than zero, then jump to the step labeled *post-cut*.
  17. If *start* is less than zero, then let *start* be zero.

18. If *position* is greater than *subpath width*, then let *end* be the offset *subpath width* on *L*. Otherwise, let *end* be the offset *position* on *L*.

19. Jump to the first appropriate step:

**If *segment length* is zero and *current state* is off**

Do nothing, just continue to the next step.

**If *current state* is off**

Cut the line on which *end* finds itself short at *end* and place a point there, cutting in two the subpath that it was in; remove all line segments, joins, points, and subpaths that are between *start* and *end*; and finally place a single point at *start* with no lines connecting to it.

The point has a *directionality* for the purposes of drawing line caps (see below). The directionality is the direction that the original line had at that point (i.e. when *L* was defined above).

**Otherwise**

Cut the line on which *start* finds itself into two at *start* and place a point there, cutting in two the subpath that it was in, and similarly cut the line on which *end* finds itself short at *end* and place a point there, cutting in two the subpath that it was in, and then remove all line segments, joins, points, and subpaths that are between *start* and *end*.

If *start* and *end* are the same point, then this results in just the line being cut in two and two points being inserted there, with nothing being removed, unless a join also happens to be at that point, in which case the join must be removed.

20. *Post-cut*: If *position* is greater than *subpath width*, then jump to the step labeled *convert*.

21. If *segment length* is greater than zero, then let *positioned-at-on-dash* be false.

22. Increment *index* by one. If it is equal to the number of entries in *style's* [dash list](#), then let *index* be 0.

23. Return to the step labeled *dash on*.

9. *Convert*: This is the step that converts the path to a new path that represents its stroke.

Create a new [path](#) that describes the edge of the areas that would be covered if a straight line of length equal to *style's* [lineWidth](#) was swept along each subpath in *path* while being kept at an angle such that the line is orthogonal to the path being swept, replacing each point with the end cap necessary to satisfy *style's* [lineCap](#) attribute as described previously and elaborated below, and replacing each join with the join necessary to satisfy *style's* [lineJoin](#) type, as defined below.



**Caps:** Each point has a flat edge perpendicular to the direction of the line coming out of it. This is then augmented according to the value of *style's* lineCap. The "butt" value means that no additional line cap is added. The "round" value means that a semi-circle with the diameter equal to *style's* lineWidth width must additionally be placed on to the line coming out of each point. The "square" value means that a rectangle with the length of *style's* lineWidth width and the width of half *style's* lineWidth width, placed flat against the edge perpendicular to the direction of the line coming out of the point, must be added at each point.

Points with no lines coming out of them must have two caps placed back-to-back as if it was really two points connected to each other by an infinitesimally short straight line in the direction of the point's *directionality* (as defined above).

**Joins:** In addition to the point where a join occurs, two additional points are relevant to each join, one for each line: the two corners found half the line width away from the join point, one perpendicular to each line, each on the side furthest from the other line.

A triangle connecting these two opposite corners with a straight line, with the third point of the triangle being the join point, must be added at all joins. The lineJoin attribute controls whether anything else is rendered. The three aforementioned values have the following meanings:

The "bevel" value means that this is all that is rendered at joins.

The "round" value means that an arc connecting the two aforementioned corners of the join, abutting (and not overlapping) the aforementioned triangle, with the diameter equal to the line width and the origin at the point of the join, must be added at joins.

The "miter" value means that a second triangle must (if it can given the miter length) be added at the join, with one line being the line between the two aforementioned corners, abutting the first triangle, and the other two being continuations of the outside edges of the two joining lines, as long as required to intersect without going over the miter length.

The miter length is the distance from the point where the join occurs to the intersection of the line edges on the outside of the join. The miter limit ratio is the maximum allowed ratio of the miter length to half the line width. If the miter length would cause the miter limit ratio (as set by *style's* miterLimit attribute) to be exceeded, then this second triangle must not be added.

The subpaths in the newly created path must be oriented such that for any point, the number of times a half-infinite straight line drawn from that point crosses a subpath is even if and only if the number of times a half-infinite straight line drawn from that same point crosses a subpath going in one direction is equal to the number of times it crosses a subpath going in the other direction.

10. Return the newly created path.

#### 4.12.5.1.4 Text styles

```
context.font [ = value ]
```

✓MDN

```
styles.font [ = value ]
```

Returns the current font settings.

Can be set, to change the font. The syntax is the same as for the CSS ['font'](#) property; values that cannot be parsed as CSS font values are ignored.

Relative keywords and lengths are computed relative to the font of the [canvas](#) element.

```
context.textAlign [ = value ]
```

✓MDN

```
styles.textAlign [ = value ]
```

Returns the current text alignment settings.

Can be set, to change the alignment. The possible values and their meanings are given below. Other values are ignored. The default is "start".

```
context.textBaseline [ = value ]
```

✓MDN

```
styles.textBaseline [ = value ]
```

Returns the current baseline alignment settings.

Can be set, to change the baseline alignment. The possible values and their meanings are given below. Other values are ignored. The default is "[alphanumeric](#)".

```
context.direction [ = value ]
```

✓MDN

```
styles.direction [ = value ]
```

Returns the current directionality.

Can be set, to change the directionality. The possible values and their meanings are given below. Other values are ignored. The default is "[inherit](#)".

```
context.letterSpacing [ = value ]
```

```
styles.letterSpacing [ = value ]
```

Returns the current spacing between characters in the text.

Can be set, to change spacing between characters. Values that cannot be parsed as a CSS [<length>](#) are ignored. The default is "0px".

```
context.fontKerning [ = value ]
```

```
styles.fontKerning [ = value ]
```

Returns the current font kerning settings.

Can be set, to change the font kerning. The possible values and their meanings are given below. Other values are ignored. The default is "[auto](#)".

```
context.fontStretch [ = value ]
```

```
styles.fontStretch [ = value ]
```

Returns the current font stretch settings.

Can be set, to change the font stretch. The possible values and their meanings are given below. Other values are ignored. The default is "[normal](#)".

```
context.fontVariantCaps [ = value ]
```

```
styles.fontVariantCaps [ = value ]
```

Returns the current font variant caps settings.

Can be set, to change the font variant caps. The possible values and their meanings are given below. Other values are ignored. The default is "[normal](#)".

```
context.textRendering [ = value ]
```

```
styles.textRendering [ = value ]
```

Returns the current text rendering settings.

Can be set, to change the text rendering. The possible values and their meanings are given below. Other values are ignored. The default is "[auto](#)".

```
context.wordSpacing [ = value ]
```

```
styles.wordSpacing [ = value ]
```

Returns the current spacing between words in the text.

Can be set, to change spacing between words. Values that cannot be parsed as a CSS [<length>](#) are ignored. The default is "0px".

Objects that implement the [CanvasTextDrawingStyles](#) interface have attributes (defined in this section) that control how text is laid out (rasterized or outlined) by the object. Such objects can also have a **font style source object**.

For [CanvasRenderingContext2D](#) objects, this is the [canvas](#) element given by the value of the context's [canvas](#) attribute.

For [OffscreenCanvasRenderingContext2D](#) objects, this is the [associated OffscreenCanvas object](#).

Font resolution for the [font style source object](#) requires a [font source](#). This is determined for a given *object* implementing [CanvasTextDrawingStyles](#) by the following steps: [\[CSSFONTLOAD\]](#)

1. If *object*'s [font style source object](#) is a [canvas](#) element, return the element's [node document](#).

2. Otherwise, *object's* [font style source object](#) is an [OffscreenCanvas](#) object:
  1. Let *global* be *object's* [relevant global object](#).
  2. If *global* is a [Window](#) object, then return *global's* [associated Document](#).
  3. [Assert](#): *global* implements [WorkerGlobalScope](#).
  4. Return *global*.

This is an example of font resolution with a regular [canvas](#) element with ID `c1`.

```
const font = new FontFace("MyCanvasFont", "url(mycanvasfont.ttf)");
documents.fonts.add(font);

const context = document.getElementById("c1").getContext("2d");
document.fonts.ready.then(function() {
  context.font = "64px MyCanvasFont";
  context.fillText("hello", 0, 0);
});
```

In this example, the canvas will display text using `mycanvasfont.ttf` as its font.

This is an example of how font resolution can happen using [OffscreenCanvas](#). Assuming a [canvas](#) element with ID `c2` which is transferred to a worker like so:

```
const offscreenCanvas =
document.getElementById("c2").transferControlToOffscreen();
worker.postMessage(offscreenCanvas, [offscreenCanvas]);
```

Then, in the worker:

```
self.onmessage = function(ev) {
  const transferredCanvas = ev.data;
  const context = transferredCanvas.getContext("2d");
  const font = new FontFace("MyFont", "url(myfont.ttf)");
  self.fonts.add(font);
  self.fonts.ready.then(function() {
    context.font = "64px MyFont";
    context.fillText("hello", 0, 0);
  });
};
```

In this example, the canvas will display a text using `myfont.ttf`. Notice that the font is only loaded inside the worker, and not in the document context.

The **font** IDL attribute, on setting, must be [parsed as a CSS <font> value](#) (but without supporting property-independent style sheet syntax like 'inherit'), and the resulting font must be assigned to the context, with the ['line-height'](#) component forced to 'normal', with the ['font-size'](#) component converted to [CSS pixels](#), and with system fonts being computed to explicit values. If the new value is syntactically incorrect (including using property-independent style sheet syntax like 'inherit' or 'initial'), then it must be ignored, without assigning a new font value. [\[CSS\]](#)

Font family names must be interpreted in the context of the [font style source object](#) when the font is to be used; any fonts embedded using `@font-face` or loaded using [FontFace](#) objects that are visible to the [font style source object](#) must therefore be available once they are loaded. (Each [font style source object](#) has a [font source](#), which determines what fonts are available.) If a font is used before it is fully loaded, or if the [font style source object](#) does not have that font in scope at the time the font is to be used, then it must be treated as if it was an unknown font, falling back to another as described by the relevant CSS specifications. [\[CSSFONTS\]](#) [\[CSSFONTLOAD\]](#)

On getting, the **font** attribute must return the [serialized form](#) of the current font of the context (with no ['line-height'](#) component). [\[CSSOM\]](#)

For example, after the following statement:

```
context.font = 'italic 400 12px/2 Unknown Font, sans-serif';
```

...the expression `context.font` would evaluate to the string `"italic 12px "Unknown Font", sans-serif"`. The "400" font-weight doesn't appear because that is the default value. The line-height doesn't appear because it is forced to "normal", the default value.

When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the font of the context must be set to 10px sans-serif. When the ['font-size'](#) component is set to lengths using percentages, ['em'](#) or ['ex'](#) units, or the 'larger' or 'smaller' keywords, these must be interpreted relative to the [computed value](#) of the ['font-size'](#) property of the [font style source object](#) at the time that the attribute is set, if it is an element. When the ['font-weight'](#) component is set to the relative values 'bolder' and 'lighter', these must be interpreted relative to the [computed value](#) of the ['font-weight'](#) property of the [font style source object](#) at the time that the attribute is set, if it is an element. If the [computed values](#) are undefined for a particular case (e.g. because the [font style source object](#) is not an element or is not [being rendered](#)), then the relative keywords must be interpreted relative to the normal-weight 10px sans-serif default.

The **textAlign** IDL attribute, on getting, must return the current value. On setting, the current value must be changed to the new value. When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the [textAlign](#) attribute must initially have the value [start](#).

The **textBaseline** IDL attribute, on getting, must return the current value. On setting, the current value must be changed to the new value. When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the [textBaseline](#) attribute must initially have the value [alphabetic](#).

The **direction** IDL attribute, on getting, must return the current value. On setting, the current value must be changed to the new value. When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the [direction](#) attribute must initially have the value ["inherit"](#).

Objects that implement the [CanvasTextDrawingStyles](#) interface have attributes that control the spacing between letters and words. Such objects have associated **letter spacing** and **word spacing** values, which are CSS [<length>](#) values. Initially, both must be the result of [parsing](#) `"0px"` as a CSS [<length>](#).



The **letterSpacing** getter steps are to return the [serialized form](#) of [this's letter spacing](#).

The [letterSpacing](#) setter steps are:

1. Let *parsed* be the result of [parsing](#) the given value as a CSS [<length>](#).
2. If *parsed* is failure, then return.
3. Set [this's letter spacing](#) to *parsed*.



The **wordSpacing** getter steps are to return the [serialized form](#) of [this's word spacing](#).

The [wordSpacing](#) setter steps are:

1. Let *parsed* be the result of [parsing](#) the given value as a CSS [<length>](#).
2. If *parsed* is failure, then return.
3. Set [this's word spacing](#) to *parsed*.



The **fontKerning** IDL attribute, on getting, must return the current value. On setting, the current value must be changed to the new value. When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the [fontKerning](#) attribute must initially have the value ["auto"](#).



The **fontStretch** IDL attribute, on getting, must return the current value. On setting, the current value must be changed to the new value. When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the [fontStretch](#) attribute must initially have the value "[normal](#)".



The **fontVariantCaps** IDL attribute, on getting, must return the current value. On setting, the current value must be changed to the new value. When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the [fontVariantCaps](#) attribute must initially have the value "[normal](#)".



The **textRendering** IDL attribute, on getting, must return the current value. On setting, the current value must be changed to the new value. When the object implementing the [CanvasTextDrawingStyles](#) interface is created, the [textRendering](#) attribute must initially have the value "[auto](#)".

The [textAlign](#) attribute's allowed keywords are as follows:

**start**

Align to the start edge of the text (left side in left-to-right text, right side in right-to-left text).

**end**

Align to the end edge of the text (right side in left-to-right text, left side in right-to-left text).

**left**

Align to the left.

**right**

Align to the right.

**center**

Align to the center.

The [textBaseline](#) attribute's allowed keywords correspond to alignment points in the font:



The keywords map to these alignment points as follows:

**top**

The top of the em square

**hanging**

The [hanging baseline](#)

**middle**

The middle of the em square

**alphabetic**

The [alphabetic baseline](#)

**ideographic**

The [ideographic-under baseline](#)

**bottom**

The bottom of the em square

The [direction](#) attribute's allowed keywords are as follows:

**ltr**

Treat input to the [text preparation algorithm](#) as left-to-right text.

**rtl**

Treat input to the [text preparation algorithm](#) as right-to-left text.

**inherit**

Default to the directionality of the [canvas](#) element or [Document](#) as appropriate.

The [fontKerning](#) attribute's allowed keywords are as follows:



**auto**

Kerning is applied at the discretion of the user agent.

**normal**

Kerning is applied.

**none**

Kerning is not applied.

The fontStretch attribute's allowed keywords are as follows:

**ultra-condensed**

Same as CSS ['font-stretch' 'ultra-condensed'](#) setting.

**extra-condensed**

Same as CSS ['font-stretch' 'extra-condensed'](#) setting.

**condensed**

Same as CSS ['font-stretch' 'condensed'](#) setting.

**semi-condensed**

Same as CSS ['font-stretch' 'semi-condensed'](#) setting.

**normal**

The default setting, where width of the glyphs is at 100%.

**semi-expanded**

Same as CSS ['font-stretch' 'semi-expanded'](#) setting.

**expanded**

Same as CSS ['font-stretch' 'expanded'](#) setting.

**extra-expanded**

Same as CSS ['font-stretch' 'extra-expanded'](#) setting.

**ultra-expanded**

Same as CSS ['font-stretch' 'ultra-expanded'](#) setting.

The fontVariantCaps attribute's allowed keywords are as follows:

**normal**

None of the features listed below are enabled.

**small-caps**

Same as CSS ['font-variant-caps' 'small-caps'](#) setting.

**all-small-caps**

Same as CSS ['font-variant-caps' 'all-small-caps'](#) setting.

#### **petite-caps**

Same as CSS ['font-variant-caps' 'petite-caps'](#) setting.

#### **all-petite-caps**

Same as CSS ['font-variant-caps' 'all-petite-caps'](#) setting.

#### **unicase**

Same as CSS ['font-variant-caps' 'unicase'](#) setting.

#### **titling-caps**

Same as CSS ['font-variant-caps' 'titling-caps'](#) setting.

The [textRendering](#) attribute's allowed keywords are as follows:

#### **auto**

Same as 'auto' in [SVG text-rendering](#) property.

#### **optimizeSpeed**

Same as 'optimizeSpeed' in [SVG text-rendering](#) property.

#### **optimizeLegibility**

Same as 'optimizeLegibility' in [SVG text-rendering](#) property.

#### **geometricPrecision**

Same as 'geometricPrecision' in [SVG text-rendering](#) property.

The **text preparation algorithm** is as follows. It takes as input a string *text*, a [CanvasTextDrawingStyles](#) object *target*, and an optional length *maxWidth*. It returns an array of glyph shapes, each positioned on a common coordinate space, a *physical alignment* whose value is one of *left*, *right*, and *center*, and an [inline box](#). (Most callers of this algorithm ignore the *physical alignment* and the [inline box](#).)

1. If *maxWidth* was provided but is less than or equal to zero or equal to NaN, then return an empty array.
2. Replace all [ASCII whitespace](#) in *text* with U+0020 SPACE characters.
3. Let *font* be the current font of *target*, as given by that object's [font](#) attribute.
4. Apply the appropriate step from the following list to determine the value of *direction*:

**If the *target* object's [direction](#) attribute has the value ["ltr"](#)**

Let *direction* be ['ltr'](#).

**If the *target* object's [direction](#) attribute has the value ["rtl"](#)**

Let *direction* be ['rtl'](#).

**If the *target* object's [font style source object](#) is an element**

Let *direction* be [the directionality](#) of the *target* object's [font style source object](#).

If the *target* object's [font style source object](#) is a [Document](#) with a non-null [document element](#)

Let *direction* be [the directionality](#) of the *target* object's [font style source object](#)'s [document element](#).

Otherwise

Let *direction* be ['ltr'](#).

5. Form a hypothetical infinitely-wide CSS [line box](#) containing a single [inline box](#) containing the text *text*, with its CSS properties set as follows:

Property	Source
<a href="#">'direction'</a>	<i>direction</i>
<a href="#">'font'</a>	<i>font</i>
<a href="#">'font-kerning'</a>	<i>target</i> 's <a href="#">fontKerning</a>
<a href="#">'font-stretch'</a>	<i>target</i> 's <a href="#">fontStretch</a>
<a href="#">'font-variant-caps'</a>	<i>target</i> 's <a href="#">fontVariantCaps</a>
<a href="#">'letter-spacing'</a>	<i>target</i> 's <a href="#">letter spacing</a>
<a href="#">SVG text-rendering</a>	<i>target</i> 's <a href="#">textRendering</a>
<a href="#">'white-space'</a>	<a href="#">'pre'</a>
<a href="#">'word-spacing'</a>	<i>target</i> 's <a href="#">word spacing</a>

6. and with all other properties set to their initial values.
7. If *maxWidth* was provided and the hypothetical width of the [inline box](#) in the hypothetical [line box](#) is greater than *maxWidth* [CSS pixels](#), then change *font* to have a more condensed font (if one is available or if a reasonably readable one can be synthesized by applying a horizontal scale factor to the font) or a smaller font, and return to the previous step.
8. The *anchor point* is a point on the [inline box](#), and the *physical alignment* is one of the values *left*, *right*, and *center*. These variables are determined by the [textAlign](#) and [textBaseline](#) values as follows:

Horizontal position:

If [textAlign](#) is [left](#)

If [textAlign](#) is [start](#) and *direction* is ['ltr'](#)

If [textAlign](#) is [end](#) and *direction* is ['rtl'](#)

Let the *anchor point*'s horizontal position be the left edge of the [inline box](#), and let *physical alignment* be *left*.

If [textAlign](#) is [right](#)

If [textAlign](#) is [end](#) and *direction* is ['ltr'](#)

If **textAlign** is **start** and ***direction*** is 'rtl'

Let the *anchor point*'s horizontal position be the right edge of the [inline box](#), and let *physical alignment* be *right*.

If **textAlign** is **center**

Let the *anchor point*'s horizontal position be half way between the left and right edges of the [inline box](#), and let *physical alignment* be *center*.

Vertical position:

If **textBaseline** is **top**

Let the *anchor point*'s vertical position be the top of the em box of the [first available font](#) of the [inline box](#).

If **textBaseline** is **hanging**

Let the *anchor point*'s vertical position be the [hanging baseline](#) of the [first available font](#) of the [inline box](#).

If **textBaseline** is **middle**

Let the *anchor point*'s vertical position be half way between the bottom and the top of the em box of the [first available font](#) of the [inline box](#).

If **textBaseline** is **alphabetic**

Let the *anchor point*'s vertical position be the [alphabetic baseline](#) of the [first available font](#) of the [inline box](#).

If **textBaseline** is **ideographic**

Let the *anchor point*'s vertical position be the [ideographic-under baseline](#) of the [first available font](#) of the [inline box](#).

If **textBaseline** is **bottom**

Let the *anchor point*'s vertical position be the bottom of the em box of the [first available font](#) of the [inline box](#).

9. Let *result* be an array constructed by iterating over each glyph in the [inline box](#) from left to right (if any), adding to the array, for each glyph, the shape of the glyph as it is in the [inline box](#), positioned on a coordinate space using [CSS pixels](#) with its origin is at the *anchor point*.
10. Return *result*, *physical alignment*, and the inline box.

#### 4.12.5.1.5 Building paths

Objects that implement the [CanvasPath](#) interface have a [path](#). A **path** has a list of zero or more subpaths. Each subpath consists of a list of one or more points, connected by straight or curved **line segments**, and a flag indicating whether the subpath is closed or not. A closed subpath is one where the last point of the subpath

is connected to the first point of the subpath by a straight line. Subpaths with only one point are ignored when painting the path.

[Paths](#) have a **need new subpath** flag. When this flag is set, certain APIs create a new subpath rather than extending the previous one. When a [path](#) is created, its [need new subpath](#) flag must be set.

When an object implementing the [CanvasPath](#) interface is created, its [path](#) must be initialized to zero subpaths.

```
context.moveTo(x, y)
```

✓MDN

```
path.moveTo(x, y)
```

Creates a new subpath with the given point.

```
context.closePath()
```

✓MDN

```
path.closePath()
```

Marks the current subpath as closed, and starts a new subpath with a point the same as the start and end of the newly closed subpath.

```
context.lineTo(x, y)
```

✓MDN

```
path.lineTo(x, y)
```

Adds the given point to the current subpath, connected to the previous one by a straight line.

```
context.quadraticCurveTo(cpx, cpy, x, y)
```

✓MDN

```
path.quadraticCurveTo(cpx, cpy, x, y)
```

Adds the given point to the current subpath, connected to the previous one by a quadratic Bézier curve with the given control point.

```
context.bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y)
```

✓MDN

```
path.bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y)
```

Adds the given point to the current subpath, connected to the previous one by a cubic Bézier curve with the given control points.

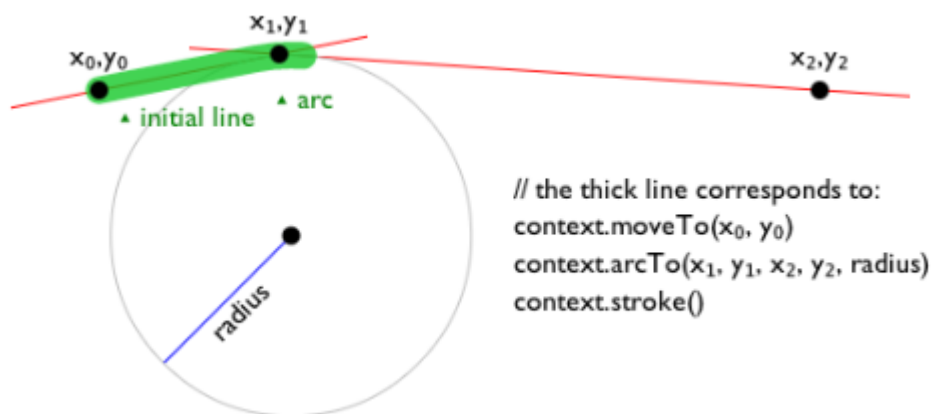
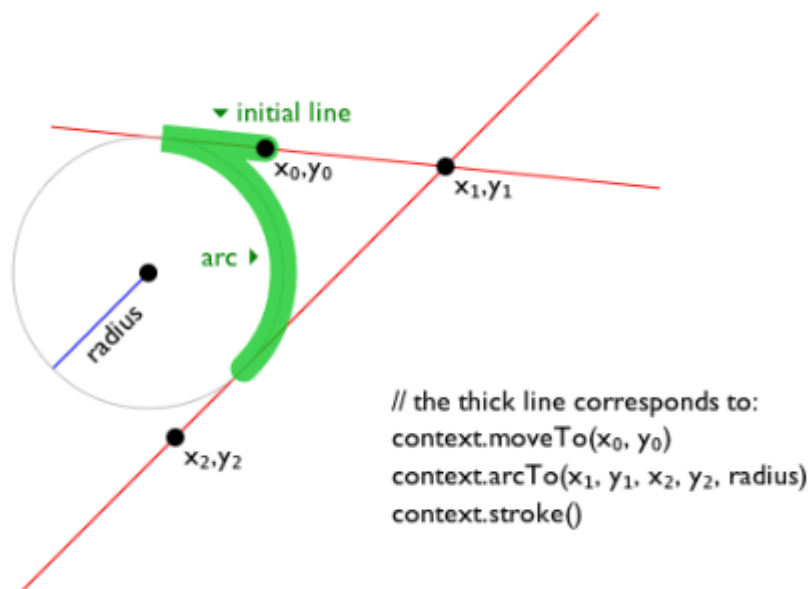
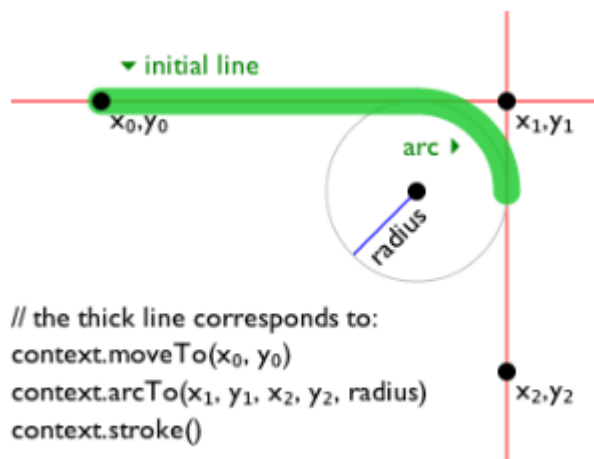
```
context.arcTo(x1, y1, x2, y2, radius)
```

✓MDN

```
path.arcTo(x1, y1, x2, y2, radius)
```

Adds an arc with the given control points and radius to the current subpath, connected to the previous point by a straight line.

Throws an ["IndexSizeError" DOMException](#) if the given radius is negative.



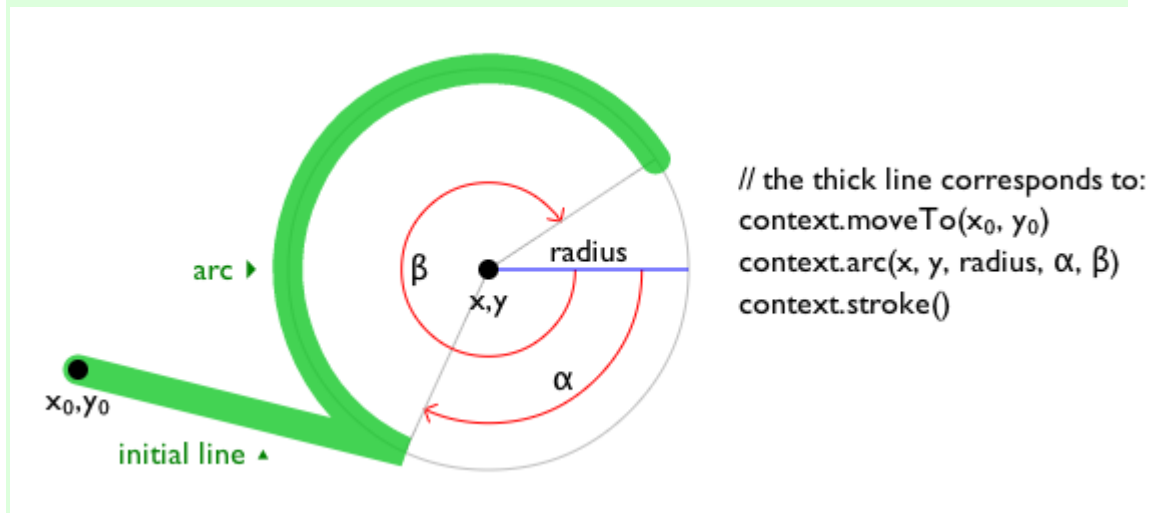
`context.arc(x, y, radius, startAngle, endAngle [, counterclockwise])`

✓MDN

`path.arc(x, y, radius, startAngle, endAngle [, counterclockwise])`

Adds points to the subpath such that the arc described by the circumference of the circle described by the arguments, starting at the given start angle and ending at the given end angle, going in the given direction (defaulting to clockwise), is added to the path, connected to the previous point by a straight line.

Throws an ["IndexSizeError" DOMException](#) if the given radius is negative.



```
context.ellipse(x, y, radiusX, radiusY, rotation, startAngle, endAngle [, counterclockwise])
```

✓MDN

```
path.ellipse(x, y, radiusX, radiusY, rotation, startAngle, endAngle [, counterclockwise])
```

Adds points to the subpath such that the arc described by the circumference of the ellipse described by the arguments, starting at the given start angle and ending at the given end angle, going in the given direction (defaulting to clockwise), is added to the path, connected to the previous point by a straight line.

Throws an ["IndexSizeError" DOMException](#) if the given radius is negative.

```
context.rect(x, y, w, h)
```

✓MDN

```
path.rect(x, y, w, h)
```

Adds a new closed subpath to the path, representing the given rectangle.

```
context.roundRect(x, y, w, h, radii)
```

```
path.roundRect(x, y, w, h, radii)
```

Adds a new closed subpath to the path representing the given rounded rectangle. *radii* is either a list of radii or a single radius representing the corners of the rectangle in pixels. If a list is provided, the number and order of these radii function in the same way as the CSS ['border-radius'](#) property. A single radius behaves the same way as a list with a single element.

If *w* and *h* are both greater than or equal to 0, or if both are smaller than 0, then the path is drawn clockwise. Otherwise, it is drawn counterclockwise.

When  $w$  is negative, the rounded rectangle is flipped horizontally, which means that the radius values that normally apply to the left corners are used on the right and vice versa. Similarly, when  $h$  is negative, the rounded rect is flipped vertically.

When a value  $r$  in *radii* is a number, the corresponding corner(s) are drawn as circular arcs of radius  $r$ .

When a value  $r$  in *radii* is an object with  $\{ x, y \}$  properties, the corresponding corner(s) are drawn as elliptical arcs whose  $x$  and  $y$  radii are equal to  $r.x$  and  $r.y$ , respectively.

When the sum of the *radii* of two corners of the same edge is greater than the length of the edge, all the *radii* of the rounded rectangle are scaled by a factor of  $\text{length} / (r1 + r2)$ . If multiple edges have this property, the scale factor of the edge with the smallest scale factor is used. This is consistent with CSS behavior.

Throws a [RangeError](#) if *radii* is a list whose size is not one, two, three, or four.

Throws a [RangeError](#) if a value in *radii* is a negative number, or is an  $\{ x, y \}$  object whose  $x$  or  $y$  properties are negative numbers.

The following methods allow authors to manipulate the [paths](#) of objects implementing the [CanvasPath](#) interface.

For objects implementing the [CanvasDrawPath](#) and [CanvasTransform](#) interfaces, the points passed to the methods, and the resulting lines added to [current default path](#) by these methods, must be transformed according to the [current transformation matrix](#) before being added to the path.

The [moveTo\( \$x\$ ,  \$y\$ \)](#) method, when invoked, must run these steps:

1. If either of the arguments are infinite or NaN, then return.
2. Create a new subpath with the specified point as its first (and only) point.

When the user agent is to **ensure there is a subpath** for a coordinate  $(x, y)$  on a [path](#), the user agent must check to see if the [path](#) has its [need new subpath](#) flag set. If it does, then the user agent must create a new subpath with the point  $(x, y)$  as its first (and only) point, as if the [moveTo\(\)](#) method had been called, and must then unset the [path](#)'s [need new subpath](#) flag.

The [closePath\(\)](#) method, when invoked, must do nothing if the object's path has no subpaths. Otherwise, it must mark the last subpath as closed, create a new subpath whose first point is the same as the previous subpath's first point, and finally add this new subpath to the path.



*If the last subpath had more than one point in its list of points, then this is equivalent to adding a straight line connecting the last point back to the first point of the last subpath, thus "closing" the subpath.*

---

New points and the lines connecting them are added to subpaths using the methods described below. In all cases, the methods only modify the last subpath in the object's path.

The `lineTo(x, y)` method, when invoked, must run these steps:

1. If either of the arguments are infinite or NaN, then return.
2. If the object's path has no subpaths, then [ensure there is a subpath](#) for (x, y).
3. Otherwise, connect the last point in the subpath to the given point (x, y) using a straight line, and then add the given point (x, y) to the subpath.

The `quadraticCurveTo(cpx, cpy, x, y)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. [Ensure there is a subpath](#) for (cpx, cpy)
3. Connect the last point in the subpath to the given point (x, y) using a quadratic Bézier curve with control point (cpx, cpy). [\[BEZIER\]](#)
4. Add the given point (x, y) to the subpath.

The `bezierCurveTo(cp1x, cp1y, cp2x, cp2y, x, y)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
  2. [Ensure there is a subpath](#) for (cp1x, cp1y).
  3. Connect the last point in the subpath to the given point (x, y) using a cubic Bézier curve with control points (cp1x, cp1y) and (cp2x, cp2y). [\[BEZIER\]](#)
  4. Add the point (x, y) to the subpath.
-

The `arcTo(x1, y1, x2, y2, radius)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. [Ensure there is a subpath](#) for (x1, y1).
3. If *radius* is negative, then throw an `"IndexSizeError" DOMException`.
4. Let the point (x0, y0) be the last point in the subpath, transformed by the inverse of the [current transformation matrix](#) (so that it is in the same coordinate system as the points passed to the method).
5. If the point (x0, y0) is equal to the point (x1, y1), or if the point (x1, y1) is equal to the point (x2, y2), or if *radius* is zero, then add the point (x1, y1) to the subpath, and connect that point to the previous point (x0, y0) by a straight line.
6. Otherwise, if the points (x0, y0), (x1, y1), and (x2, y2) all lie on a single straight line, then add the point (x1, y1) to the subpath, and connect that point to the previous point (x0, y0) by a straight line.
7. Otherwise, let *The Arc* be the shortest arc given by circumference of the circle that has radius *radius*, and that has one point tangent to the half-infinite line that crosses the point (x0, y0) and ends at the point (x1, y1), and that has a different point tangent to the half-infinite line that ends at the point (x1, y1) and crosses the point (x2, y2). The points at which this circle touches these two lines are called the start and end tangent points respectively. Connect the point (x0, y0) to the start tangent point by a straight line, adding the start tangent point to the subpath, and then connect the start tangent point to the end tangent point by *The Arc*, adding the end tangent point to the subpath.

---

The `arc(x, y, radius, startAngle, endAngle, counterclockwise)` method, when invoked, must run the [ellipse method steps](#) with this, x, y, radius, radius, 0, startAngle, endAngle, and counterclockwise.

*This makes it equivalent to `ellipse()` except that both radii are equal and rotation is 0.*

The `ellipse(x, y, radiusX, radiusY, rotation, startAngle, endAngle, counterclockwise)` method, when invoked, must run the [ellipse method steps](#) with this, x, y, radiusX, radiusY, rotation, startAngle, endAngle, and counterclockwise.

The **ellipse method steps**, given *canvasPath*, x, y, radiusX, radiusY, rotation, startAngle, endAngle, and counterclockwise, are:

1. If any of the arguments are infinite or NaN, then return.
2. If either *radiusX* or *radiusY* are negative, then throw an `"IndexSizeError" DOMException`.
3. If *canvasPath*'s path has any subpaths, then add a straight line from the last point in the subpath to the start point of the arc.
4. Add the start and end points of the arc to the subpath, and connect them with an arc. The arc and its start and end points are defined as follows:

Consider an ellipse that has its origin at  $(x, y)$ , that has a major-axis radius *radiusX* and a minor-axis radius *radiusY*, and that is rotated about its origin such that its semi-major axis is inclined *rotation* radians clockwise from the x-axis.

If *counterclockwise* is false and *endAngle*-*startAngle* is equal to or greater than  $2\pi$ , or, if *counterclockwise* is true and *startAngle*-*endAngle* is equal to or greater than  $2\pi$ , then the arc is the whole circumference of this ellipse, and the point at *startAngle* along this circle's circumference, measured in radians clockwise from the ellipse's semi-major axis, acts as both the start point and the end point.

Otherwise, the points at *startAngle* and *endAngle* along this circle's circumference, measured in radians clockwise from the ellipse's semi-major axis, are the start and end points respectively, and the arc is the path along the circumference of this ellipse from the start point to the end point, going counterclockwise if *counterclockwise* is true, and clockwise otherwise. Since the points are on the ellipse, as opposed to being simply angles from zero, the arc can never cover an angle greater than  $2\pi$  radians.

*Even if the arc covers the entire circumference of the ellipse and there are no other points in the subpath, the path is not closed unless the `closePath()` method is appropriately invoked.*

---

The `rect(x, y, w, h)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. Create a new subpath containing just the four points  $(x, y)$ ,  $(x+w, y)$ ,  $(x+w, y+h)$ ,  $(x, y+h)$ , in that order, with those four points connected by straight lines.
3. Mark the subpath as closed.
4. Create a new subpath with the point  $(x, y)$  as the only point in the subpath.

The `roundRect(x, y, w, h, radii)` method steps are:

1. If any of *x*, *y*, *w*, or *h* are infinite or NaN, then return.
2. If *radii* is an [unrestricted double](#) or [DOMPointInit](#), then set *radii* to « *radii* ».
3. If *radii* is not a list of size one, two, three, or four, then throw a [RangeError](#).
4. Let *normalizedRadii* be an empty list.
5. For each *radius* of *radii*:
  1. If *radius* is a [DOMPointInit](#):
    1. If *radius*["[x](#)"] or *radius*["[y](#)"] is infinite or NaN, then return.
    2. If *radius*["[x](#)"] or *radius*["[y](#)"] is negative, then throw a [RangeError](#).
    3. Otherwise, append *radius* to *normalizedRadii*.
  2. If *radius* is a [unrestricted double](#):
    1. If *radius* is infinite or NaN, then return.
    2. If *radius* is negative, then throw a [RangeError](#).
    3. Otherwise append «["[x](#)" → *radius*, "[y](#)" → *radius*]» to *normalizedRadii*.
6. Let *upperLeft*, *upperRight*, *lowerRight*, and *lowerLeft* be null.
7. If *normalizedRadii*'s size is 4, then set *upperLeft* to *normalizedRadii*[0], set *upperRight* to *normalizedRadii*[1], set *lowerRight* to *normalizedRadii*[2], and set *lowerLeft* to *normalizedRadii*[3].
8. If *normalizedRadii*'s size is 3, then set *upperLeft* to *normalizedRadii*[0], set *upperRight* and *lowerLeft* to *normalizedRadii*[1], and set *lowerRight* to *normalizedRadii*[2].
9. If *normalizedRadii*'s size is 2, then set *upperLeft* and *lowerRight* to *normalizedRadii*[0] and set *upperRight* and *lowerLeft* to *normalizedRadii*[1].
10. If *normalizedRadii*'s size is 1, then set *upperLeft*, *upperRight*, *lowerRight*, and *lowerLeft* to *normalizedRadii*[0].
11. Corner curves must not overlap. Scale all radii to prevent this:

1. Let *top* be  $upperLeft["\underline{x}"] + upperRight["\underline{x}"]$ .
2. Let *right* be  $upperRight["\underline{y}"] + lowerRight["\underline{y}"]$ .
3. Let *bottom* be  $lowerRight["\underline{x}"] + lowerLeft["\underline{x}"]$ .
4. Let *left* be  $upperLeft["\underline{y}"] + lowerLeft["\underline{y}"]$ .
5. Let *scale* be the minimum value of the ratios  $w / top$ ,  $h / right$ ,  $w / bottom$ ,  $h / left$ .
6. If *scale* is less than 1, then set the  $\underline{x}$  and  $\underline{y}$  members of *upperLeft*, *upperRight*, *lowerLeft*, and *lowerRight* to their current values multiplied by *scale*.

12. Create a new subpath:

1. Move to the point  $(x + upperLeft["\underline{x}"], y)$ .
2. Draw a straight line to the point  $(x + w - upperRight["\underline{x}"], y)$ .
3. Draw an arc to the point  $(x + w, y + upperRight["\underline{y}"])$ .
4. Draw a straight line to the point  $(x + w, y + h - lowerRight["\underline{y}"])$ .
5. Draw an arc to the point  $(x + w - lowerRight["\underline{x}"], y + h)$ .
6. Draw a straight line to the point  $(x + lowerLeft["\underline{x}"], y + h)$ .
7. Draw an arc to the point  $(x, y + h - lowerLeft["\underline{y}"])$ .
8. Draw a straight line to the point  $(x, y + upperLeft["\underline{y}"])$ .
9. Draw an arc to the point  $(x + upperLeft["\underline{x}"], y)$ .

13. Mark the subpath as closed.

14. Create a new subpath with the point  $(x, y)$  as the only point in the subpath.

*This is designed to behave similarly to the CSS ['border-radius'](#) property.*

#### 4.12.5.1.6 [Path2D](#) objects



[Path2D](#) objects can be used to declare paths that are then later used on objects implementing the [CanvasDrawPath](#) interface. In addition to many of the APIs

described in earlier sections, [Path2D](#) objects have methods to combine paths, and to add text to paths.

```
path = new Path2D()
```

✓MDN

Creates a new empty [Path2D](#) object.

```
path = new Path2D(path)
```

When *path* is a [Path2D](#) object, returns a copy.

When *path* is a string, creates the path described by the argument, interpreted as SVG path data. [\[SVG\]](#)

```
path.addPath(path [, transform ])
```

✓MDN

Adds to the path the path given by the argument.

The [Path2D](#)(*path*) constructor, when invoked, must run these steps:

1. Let *output* be a new [Path2D](#) object.
2. If *path* is not given, then return *output*.
3. If *path* is a [Path2D](#) object, then add all subpaths of *path* to *output* and return *output*. (In other words, it returns a copy of the argument.)
4. Let *svgPath* be the result of parsing and interpreting *path* according to SVG 2's rules for path data. [\[SVG\]](#)

*The resulting path could be empty. SVG defines error handling rules for parsing and applying path data.*

5. Let (*x*, *y*) be the last point in *svgPath*.
6. Add all the subpaths, if any, from *svgPath* to *output*.
7. Create a new subpath in *output* with (*x*, *y*) as the only point in the subpath.
8. Return *output*.

---

The [addPath](#)(*path*, *transform*) method, when invoked on a [Path2D](#) object *a*, must run these steps:

1. If the [Path2D](#) object *path* has no subpaths, then return.

2. Let *matrix* be the result of [creating a DOMMatrix from the 2D dictionary transform](#).
3. If one or more of *matrix*'s [m11 element](#), [m12 element](#), [m21 element](#), [m22 element](#), [m41 element](#), or [m42 element](#) are infinite or NaN, then return.
4. Create a copy of all the subpaths in *path*. Let this copy be known as *c*.
5. Transform all the coordinates and lines in *c* by the transform matrix *matrix*.
6. Let (*x*, *y*) be the last point in the last subpath of *c*.
7. Add all the subpaths in *c* to *a*.
8. Create a new subpath in *a* with (*x*, *y*) as the only point in the subpath.

#### 4.12.5.1.7 Transformations

Objects that implement the [CanvasTransform](#) interface have a **current transformation matrix**, as well as methods (described in this section) to manipulate it. When an object implementing the [CanvasTransform](#) interface is created, its transformation matrix must be initialized to the identity matrix.

The [current transformation matrix](#) is applied to coordinates when creating the [current default path](#), and when painting text, shapes, and [Path2D](#) objects, on objects implementing the [CanvasTransform](#) interface.

The transformations must be performed in reverse order.

*For instance, if a scale transformation that doubles the width is applied to the canvas, followed by a rotation transformation that rotates drawing operations by a quarter turn, and a rectangle twice as wide as it is tall is then drawn on the canvas, the actual result will be a square.*

```
context.scale(x, y)
```

✓MDN

Changes the [current transformation matrix](#) to apply a scaling transformation with the given characteristics.

```
context.rotate(angle)
```

✓MDN

Changes the [current transformation matrix](#) to apply a rotation transformation with the given characteristics. The angle is in radians.

```
context.translate(x, y)
```



Changes the [current transformation matrix](#) to apply a translation transformation with the given characteristics.

```
context.transform(a, b, c, d, e, f)
```



Changes the [current transformation matrix](#) to apply the matrix given by the arguments as described below.

```
matrix = context.getTransform()
```



Returns a copy of the [current transformation matrix](#), as a newly created [DOMMatrix](#) object.

```
context.setTransform(a, b, c, d, e, f)
```



Changes the [current transformation matrix](#) to the matrix given by the arguments as described below.

```
context.setTransform(transform)
```

Changes the [current transformation matrix](#) to the matrix represented by the passed [DOMMatrix2DInit](#) dictionary.

```
context.resetTransform()
```



Changes the [current transformation matrix](#) to the identity matrix.

The `scale(x, y)` method, when invoked, must run these steps:

1. If either of the arguments are infinite or NaN, then return.
2. Add the scaling transformation described by the arguments to the [current transformation matrix](#). The `x` argument represents the scale factor in the horizontal direction and the `y` argument represents the scale factor in the vertical direction. The factors are multiples.

The `rotate(angle)` method, when invoked, must run these steps:

1. If `angle` is infinite or NaN, then return.
2. Add the rotation transformation described by the argument to the [current transformation matrix](#). The `angle` argument represents a clockwise rotation angle expressed in radians.

The `translate(x, y)` method, when invoked, must run these steps:

1. If either of the arguments are infinite or NaN, then return.



2. Add the translation transformation described by the arguments to the [current transformation matrix](#). The *x* argument represents the translation distance in the horizontal direction and the *y* argument represents the translation distance in the vertical direction. The arguments are in coordinate space units.

The `transform(a, b, c, d, e, f)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. Replace the [current transformation matrix](#) with the result of multiplying the current transformation matrix with the matrix described by:

$$\begin{matrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{matrix}$$

*The arguments *a*, *b*, *c*, *d*, *e*, and *f* are sometimes called *m11*, *m12*, *m21*, *m22*, *dx*, and *dy* or *m11*, *m21*, *m12*, *m22*, *dx*, and *dy*. Care ought to be taken in particular with the order of the second and third arguments (*b* and *c*) as their order varies from API to API and APIs sometimes use the notation *m12/m21* and sometimes *m21/m12* for those positions.*

The `getTransform()` method, when invoked, must return a newly created [DOMMatrix](#) representing a copy of the [current transformation matrix](#) matrix of the context.

*This returned object is not live, so updating it will not affect the [current transformation matrix](#), and updating the [current transformation matrix](#) will not affect an already returned [DOMMatrix](#).*

The `setTransform(a, b, c, d, e, f)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. Reset the [current transformation matrix](#) to the identity matrix.
3. Invoke the `transform(a, b, c, d, e, f)` method with the same arguments.

The `setTransform(transform)` method, when invoked, must run these steps:

1. Let *matrix* be the result of [creating a DOMMatrix from the 2D dictionary transform](#).
2. If one or more of *matrix*'s [m11 element](#), [m12 element](#), [m21 element](#), [m22 element](#), [m41 element](#), or [m42 element](#) are infinite or NaN, then return.
3. Reset the [current transformation matrix](#) to *matrix*.

The `resetTransform()` method, when invoked, must reset the [current transformation matrix](#) to the identity matrix.

Given a matrix of the form created by the `transform()` and `setTransform()` methods, i.e.,

$$\begin{pmatrix} a & c & e \\ b & d & f \\ 0 & 0 & 1 \end{pmatrix}$$

the resulting transformed coordinates after transform matrix multiplication will be

$$\begin{aligned} x_{new} &= a x + c y + e \\ y_{new} &= b x + d y + f \end{aligned}$$

#### 4.12.5.1.8 Image sources for 2D rendering contexts

Some methods on the `CanvasDrawImage` and `CanvasFillStrokeStyles` interfaces take the union type `CanvasImageSource` as an argument.

This union type allows objects implementing any of the following interfaces to be used as image sources:

- `HTMLOrSVGImageElement` (`img` or `SVG image` elements)
- `HTMLVideoElement` (`video` elements)
- `HTMLCanvasElement` (`canvas` elements)
- `ImageBitmap`
- `VideoFrame`

Although not formally specified as such, `SVG image` elements are expected to be implemented nearly identical to `img` elements. That is, `SVG image` elements share the fundamental concepts and features of `img` elements.

The `ImageBitmap` interface can be created from a number of other image-representing types, including `ImageData`.

To check the usability of the *image* argument, where *image* is a `CanvasImageSource` object, run these steps:

1. Switch on *image*:

**`HTMLOrSVGImageElement`**

If *image*'s `current request`'s `state` is `broken`, then throw an `"InvalidStateError"` `DOMException`.

If *image* is not [fully decodable](#), then return *bad*.

If *image* has an [intrinsic width](#) or [intrinsic height](#) (or both) equal to zero, then return *bad*.

#### HTMLVideoElement

If *image*'s [readyState](#) attribute is either [HAVE NOTHING](#) or [HAVE METADATA](#), then return *bad*.

#### HTMLCanvasElement

##### OffscreenCanvas

If *image* has either a horizontal dimension or a vertical dimension equal to zero, then throw an ["InvalidStateError"](#) [DOMException](#).

#### ImageBitmap

##### VideoFrame

If *image*'s [\[\[Detached\]\]](#) internal slot value is set to true, then throw an ["InvalidStateError"](#) [DOMException](#).

2. Return *good*.

When a [CanvasImageSource](#) object represents an [HTMLOrSVGImageElement](#), the element's image must be used as the source image.

Specifically, when a [CanvasImageSource](#) object represents an animated image in an [HTMLOrSVGImageElement](#), the user agent must use the default image of the animation (the one that the format defines is to be used when animation is not supported or is disabled), or, if there is no such image, the first frame of the animation, when rendering the image for [CanvasRenderingContext2D](#) APIs.

When a [CanvasImageSource](#) object represents an [HTMLVideoElement](#), then the frame at the [current playback position](#) when the method with the argument is invoked must be used as the source image when rendering the image for [CanvasRenderingContext2D](#) APIs, and the source image's dimensions must be the [intrinsic width](#) and [intrinsic height](#) of the [media resource](#) (i.e., after any aspect-ratio correction has been applied).

When a [CanvasImageSource](#) object represents an [HTMLCanvasElement](#), the element's bitmap must be used as the source image.

When a [CanvasImageSource](#) object represents an element that is [being rendered](#) and that element has been resized, the original image data of the source image must be used, not the image as it is rendered (e.g. [width](#) and [height](#) attributes on the source element have no effect on how the object is interpreted when rendering the image for [CanvasRenderingContext2D](#) APIs).

When a [CanvasImageSource](#) object represents an [ImageBitmap](#), the object's bitmap image data must be used as the source image.

When a [CanvasImageSource](#) object represents a [VideoFrame](#), the object's pixel data must be used as the source image, and the source image's dimensions must be the object's [\[\[display width\]\]](#) and [\[\[display height\]\]](#).

An object *image* is **not origin-clean** if, switching on *image*'s type:

#### [HTMLImageElement](#)

*image*'s [current request](#)'s [image data](#) is [CORS-cross-origin](#).

#### [HTMLVideoElement](#)

*image*'s [media data](#) is [CORS-cross-origin](#).

#### [HTMLCanvasElement](#)

##### [ImageBitmap](#)

*image*'s bitmap's [origin-clean](#) flag is false.

#### 4.12.5.1.9 Fill and stroke styles

```
context.fillStyle [ = value ]
```



Returns the current style used for filling shapes.

Can be set, to change the [fill style](#).

The style can be either a string containing a CSS color, or a [CanvasGradient](#) or [CanvasPattern](#) object. Invalid values are ignored.

```
context.strokeStyle [ = value ]
```



Returns the current style used for stroking shapes.

Can be set, to change the [stroke style](#).

The style can be either a string containing a CSS color, or a [CanvasGradient](#) or [CanvasPattern](#) object. Invalid values are ignored.

Objects that implement the [CanvasFillStrokeStyles](#) interface have attributes and methods (defined in this section) that control how shapes are treated by the object.

Such objects have associated **fill style** and **stroke style** values, which are either CSS colors, [CanvasPatterns](#), or [CanvasGradients](#). Initially, both must be the result of [parsing](#) the string "#000000".

When the value is a CSS color, it must not be affected by the transformation matrix when used to draw on bitmaps.

*When set to a [CanvasPattern](#) or [CanvasGradient](#) object, changes made to the object after the assignment do affect subsequent stroking or filling of shapes.*

The **fillStyle** getter steps are:

1. If [this](#)'s [fill style](#) is a CSS color, then return the [serialization](#) of that color.
2. Return [this](#)'s [fill style](#).

The **fillStyle** setter steps are:

1. If the given value is a string, then:
  1. Let *parsedValue* be the result of [parsing](#) the given value with [this](#)'s [canvas](#) attribute's value.
  2. If *parsedValue* is failure, then return.
  3. Set [this](#)'s [fill style](#) to *parsedValue*.
  4. Return.
2. If the given value is a [CanvasPattern](#) object that is marked as [not origin-clean](#), then set [this](#)'s [origin-clean](#) flag to false.
3. Set [this](#)'s [fill style](#) to the given value.

The **strokeStyle** getter steps are:

1. If [this](#)'s [stroke style](#) is a CSS color, then return the [serialization](#) of that color.
2. Return [this](#)'s [stroke style](#).

The **strokeStyle** setter steps are:

1. If the given value is a string, then:
  1. Let *parsedValue* be the result of [parsing](#) the given value with [this](#)'s [canvas](#) attribute's value.
  2. If *parsedValue* is failure, then return.
  3. Set [this](#)'s [stroke style](#) to *parsedValue*.
  4. Return.
2. If the given value is a [CanvasPattern](#) object that is marked as [not origin-clean](#), then set [this](#)'s [origin-clean](#) flag to false.

3. Set [this](#)'s [stroke style](#) to the given value.

The **serialization of a color** for a color value is a string, computed as follows: if it has alpha equal to 1.0, then the string is a lowercase six-digit hex value, prefixed with a "#" character (U+0023 NUMBER SIGN), with the first two digits representing the red component, the next two digits representing the green component, and the last two digits representing the blue component, the digits being [ASCII lower hex digits](#). Otherwise, the color value has alpha less than 1.0, and the string is the color value in the CSS `rgba()` functional-notation format: the literal string "rgba" (U+0072 U+0067 U+0062 U+0061) followed by a U+0028 LEFT PARENTHESIS, a base-ten integer in the range 0-255 representing the red component (using [ASCII digits](#) in the shortest form possible), a literal U+002C COMMA and U+0020 SPACE, an integer for the green component, a comma and a space, an integer for the blue component, another comma and space, a U+0030 DIGIT ZERO, if the alpha value is greater than zero then a U+002E FULL STOP (representing the decimal point), if the alpha value is greater than zero then one or more [ASCII digits](#) representing the fractional part of the alpha, and finally a U+0029 RIGHT PARENTHESIS. User agents must express the fractional part of the alpha value, if any, with the level of precision necessary for the alpha value, when reparsed, to be interpreted as the same alpha value.

---

There are three types of gradients, linear gradients, radial gradients, and conic gradients, represented by objects implementing the opaque [CanvasGradient](#) interface.

Once a gradient has been created (see below), stops are placed along it to define how the colors are distributed along the gradient. The color of the gradient at each stop is the color specified for that stop. Between each such stop, the colors and the alpha component must be linearly interpolated over the RGBA space without premultiplying the alpha value to find the color to use at that offset. Before the first stop, the color must be the color of the first stop. After the last stop, the color must be the color of the last stop. When there are no stops, the gradient is [transparent black](#).

```
gradient.addColorStop(offset, color)
```



Adds a color stop with the given color to the gradient at the given offset. 0.0 is the offset at one end of the gradient, 1.0 is the offset at the other end.

Throws an ["IndexSizeError" DOMException](#) if the offset is out of range.

Throws a ["SyntaxError" DOMException](#) if the color cannot be parsed.

```
gradient = context.createLinearGradient(x0, y0, x1, y1)
```



Returns a [CanvasGradient](#) object that represents a linear gradient that paints along the line given by the coordinates represented by the arguments.

```
gradient = context.createRadialGradient(x0, y0, r0, x1, y1, r1)
```



Returns a [CanvasGradient](#) object that represents a radial gradient that paints along the cone given by the circles represented by the arguments.

If either of the radii are negative, throws an ["IndexSizeError" DOMException](#) exception.

```
gradient = context.createConicGradient(startAngle, x, y)
```



Returns a [CanvasGradient](#) object that represents a conic gradient that paints clockwise along the rotation around the center represented by the arguments.

The `addColorStop(offset, color)` method on the [CanvasGradient](#), when invoked, must run these steps:

1. If the *offset* is less than 0 or greater than 1, then throw an ["IndexSizeError" DOMException](#).
2. Let *parsed color* be the result of [parsing](#) *color*.

*No element is passed to the parser because [CanvasGradient](#) objects are [canvas-neutral](#) — a [CanvasGradient](#) object created by one [canvas](#) can be used by another, and there is therefore no way to know which is the "element in question" at the time that the color is specified.*

3. If *parsed color* is failure, throw a ["SyntaxError" DOMException](#).
4. Place a new stop on the gradient, at offset *offset* relative to the whole gradient, and with the color *parsed color*.

If multiple stops are added at the same offset on a gradient, then they must be placed in the order added, with the first one closest to the start of the gradient, and each subsequent one infinitesimally further along towards the end point (in effect causing all but the first and last stop added at each point to be ignored).

The `createLinearGradient(x0, y0, x1, y1)` method takes four arguments that represent the start point (x0, y0) and end point (x1, y1) of the gradient. The method, when invoked, must return a linear [CanvasGradient](#) initialized with the specified line.

Linear gradients must be rendered such that all points on a line perpendicular to the line that crosses the start and end points have the color at the point where those two lines cross (with the colors coming from the [interpolation and extrapolation](#) described above). The points in the linear gradient must be transformed as described by the [current transformation matrix](#) when rendering.

If  $x_0 = x_1$  and  $y_0 = y_1$ , then the linear gradient must paint nothing.

The `createRadialGradient( $x_0$ ,  $y_0$ ,  $r_0$ ,  $x_1$ ,  $y_1$ ,  $r_1$ )` method takes six arguments, the first three representing the start circle with origin  $(x_0, y_0)$  and radius  $r_0$ , and the last three representing the end circle with origin  $(x_1, y_1)$  and radius  $r_1$ . The values are in coordinate space units. If either of  $r_0$  or  $r_1$  are negative, then an `"IndexSizeError" DOMException` must be thrown. Otherwise, the method, when invoked, must return a radial `CanvasGradient` initialized with the two specified circles.

Radial gradients must be rendered by following these steps:

1. If  $x_0 = x_1$  and  $y_0 = y_1$  and  $r_0 = r_1$ , then the radial gradient must paint nothing. Return.

2. Let  $x(\omega) = (x_1 - x_0)\omega + x_0$

Let  $y(\omega) = (y_1 - y_0)\omega + y_0$

Let  $r(\omega) = (r_1 - r_0)\omega + r_0$

Let the color at  $\omega$  be the color at that position on the gradient (with the colors coming from the [interpolation and extrapolation](#) described above).

3. For all values of  $\omega$  where  $r(\omega) > 0$ , starting with the value of  $\omega$  nearest to positive infinity and ending with the value of  $\omega$  nearest to negative infinity, draw the circumference of the circle with radius  $r(\omega)$  at position  $(x(\omega), y(\omega))$ , with the color at  $\omega$ , but only painting on the parts of the bitmap that have not yet been painted on by earlier circles in this step for this rendering of the gradient.

*This effectively creates a cone, touched by the two circles defined in the creation of the gradient, with the part of the cone before the start circle (0.0) using the color of the first offset, the part of the cone after the end circle (1.0) using the color of the last offset, and areas outside the cone untouched by the gradient ([transparent black](#)).*

The resulting radial gradient must then be transformed as described by the [current transformation matrix](#) when rendering.

The `createConicGradient( $startAngle$ ,  $x$ ,  $y$ )` method takes three arguments, the first argument, `startAngle`, represents the angle in radians at which the gradient begins, and the last two arguments,  $(x, y)$ , represent the center of the gradient in [CSS pixels](#). The method, when invoked, must return a conic `CanvasGradient` initialized with the specified center and angle.

It follows the same rendering rule as CSS '[conic-gradient](#)' and it is equivalent to CSS '`conic-gradient(from  $adjustedStartAngle$ rad at  $xpx$   $ypx$ ,  $angularColorStopList$ )`'. Here:

- $adjustedStartAngle$  is given by  $startAngle + \pi/2$ ;



- *angularColorStopList* is given by the color stops that have been added to the [CanvasGradient](#) using [addColorStop\(\)](#), with the color stop offsets interpreted as percentages.

Gradients must be painted only where the relevant stroking or filling effects requires that they be drawn.

---

Patterns are represented by objects implementing the opaque [CanvasPattern](#) interface.

```
pattern = context.createPattern(image, repetition)
```

✓MDN

Returns a [CanvasPattern](#) object that uses the given image and repeats in the direction(s) given by the *repetition* argument.

The allowed values for *repetition* are `repeat` (both directions), `repeat-x` (horizontal only), `repeat-y` (vertical only), and `no-repeat` (neither). If the *repetition* argument is empty, the value `repeat` is used.

If the image isn't yet fully decoded, then nothing is drawn. If the image is a canvas with no data, throws an ["InvalidStateError"](#) [DOMException](#).

```
pattern.setTransform(transform)
```

✓MDN

Sets the transformation matrix that will be used when rendering the pattern during a fill or stroke painting operation.

The [createPattern\(\*image\*, \*repetition\*\)](#) method, when invoked, must run these steps:

1. Let *usability* be the result of [checking the usability of](#) *image*.
2. If *usability* is *bad*, then return null.
3. [Assert](#): *usability* is good.
4. If *repetition* is the empty string, then set it to `"repeat"`.
5. If *repetition* is not [identical to](#) one of `"repeat"`, `"repeat-x"`, `"repeat-y"`, or `"no-repeat"`, then throw a ["SyntaxError"](#) [DOMException](#).
6. Let *pattern* be a new [CanvasPattern](#) object with the image *image* and the repetition behavior given by *repetition*.
7. If *image* [is not origin-clean](#), then mark *pattern* as **not origin-clean**.

8. Return *pattern*.

Modifying the *image* used when creating a `CanvasPattern` object after calling the `createPattern()` method must not affect the pattern(s) rendered by the `CanvasPattern` object.

Patterns have a transformation matrix, which controls how the pattern is used when it is painted. Initially, a pattern's transformation matrix must be the identity matrix.

The `setTransform(transform)` method, when invoked, must run these steps:

1. Let *matrix* be the result of [creating a DOMMatrix from the 2D dictionary transform](#).
2. If one or more of *matrix*'s [m11 element](#), [m12 element](#), [m21 element](#), [m22 element](#), [m41 element](#), or [m42 element](#) are infinite or NaN, then return.
3. Reset the pattern's transformation matrix to *matrix*.

When a pattern is to be rendered within an area, the user agent must run the following steps to determine what is rendered:

1. Create an infinite [transparent black](#) bitmap.
2. Place a copy of the image on the bitmap, anchored such that its top left corner is at the origin of the coordinate space, with one coordinate space unit per [CSS pixel](#) of the image, then place repeated copies of this image horizontally to the left and right, if the repetition behavior is "repeat-x", or vertically up and down, if the repetition behavior is "repeat-y", or in all four directions all over the bitmap, if the repetition behavior is "repeat".

If the original image data is a bitmap image, then the value painted at a point in the area of the repetitions is computed by filtering the original image data. When scaling up, if the `imageSmoothingEnabled` attribute is set to false, then the image must be rendered using nearest-neighbor interpolation. Otherwise, the user agent may use any filtering algorithm (for example bilinear interpolation or nearest-neighbor). User agents which support multiple filtering algorithms may use the value of the `imageSmoothingQuality` attribute to guide the choice of filtering algorithm. When such a filtering algorithm requires a pixel value from outside the original image data, it must instead use the value from wrapping the pixel's coordinates to the original image's dimensions. (That is, the filter uses 'repeat' behavior, regardless of the value of the pattern's repetition behavior.)

3. Transform the resulting bitmap according to the pattern's transformation matrix.
4. Transform the resulting bitmap again, this time according to the [current transformation matrix](#).

5. Replace any part of the image outside the area in which the pattern is to be rendered with [transparent black](#).
  6. The resulting bitmap is what is to be rendered, with the same origin and same scale.
- 

If a radial gradient or repeated pattern is used when the transformation matrix is singular, then the resulting style must be [transparent black](#) (otherwise the gradient or pattern would be collapsed to a point or line, leaving the other pixels undefined). Linear gradients and solid colors always define all points even with singular transformation matrices.

#### 4.12.5.1.10 Drawing rectangles to the bitmap

Objects that implement the [CanvasRect](#) interface provide the following methods for immediately drawing rectangles to the bitmap. The methods each take four arguments; the first two give the  $x$  and  $y$  coordinates of the top left of the rectangle, and the second two give the width  $w$  and height  $h$  of the rectangle, respectively.

The [current transformation matrix](#) must be applied to the following four coordinates, which form the path that must then be closed to get the specified rectangle:  $(x, y)$ ,  $(x+w, y)$ ,  $(x+w, y+h)$ ,  $(x, y+h)$ .

Shapes are painted without affecting the [current default path](#), and are subject to the [clipping region](#), and, with the exception of [clearRect\(\)](#), also [shadow effects](#), [global alpha](#), and the [current compositing and blending operator](#).

```
context.clearRect(x, y, w, h)
```

✓MDN

Clears all pixels on the bitmap in the given rectangle to [transparent black](#).

```
context.fillRect(x, y, w, h)
```

✓MDN

Paints the given rectangle onto the bitmap, using the current fill style.

```
context.strokeRect(x, y, w, h)
```

✓MDN

Paints the box that outlines the given rectangle onto the bitmap, using the current stroke style.

The `clearRect(x, y, w, h)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. Let *pixels* be the set of pixels in the specified rectangle that also intersect the current [clipping region](#).
3. Clear the pixels in *pixels* to a [transparent black](#), erasing any previous image.

*If either height or width are zero, this method has no effect, since the set of pixels would be empty.*

The `fillRect(x, y, w, h)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. If either *w* or *h* are zero, then return.
3. Paint the specified rectangular area using [this's fill style](#).

The `strokeRect(x, y, w, h)` method, when invoked, must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. Take the result of [tracing the path](#) described below, using the [CanvasPathDrawingStyles](#) interface's line styles, and fill it with [this's stroke style](#).

If both *w* and *h* are zero, the path has a single subpath with just one point (*x*, *y*), and no lines, and this method thus has no effect (the [trace a path](#) algorithm returns an empty path in that case).

If just one of either *w* or *h* is zero, then the path has a single subpath consisting of two points, with coordinates (*x*, *y*) and (*x*+*w*, *y*+*h*), in that order, connected by a single straight line.

Otherwise, the path has a single subpath consisting of four points, with coordinates (*x*, *y*), (*x*+*w*, *y*), (*x*+*w*, *y*+*h*), and (*x*, *y*+*h*), connected to each other in that order by straight lines.

#### 4.12.5.1.11 Drawing text to the bitmap



```
context.fillText(text, x, y [, maxWidth ])
```



```
context.strokeText(text, x, y [, maxWidth ])
```

✓MDN

Fills or strokes (respectively) the given text at the given position. If a maximum width is provided, the text will be scaled to fit that width if necessary.

```
metrics = context.measureText(text)
```

✓MDN

Returns a [TextMetrics](#) object with the metrics of the given text in the current font.

```
metrics.width
```

✓MDN

```
metrics.actualBoundingBoxLeft
```

✓MDN

```
metrics.actualBoundingBoxRight
```

✓MDN

```
metrics.fontBoundingBoxAscent
```

✓MDN

```
metrics.fontBoundingBoxDescent
```

✓MDN

```
metrics.actualBoundingBoxAscent
```

✓MDN

```
metrics.actualBoundingBoxDescent
```

✓MDN

```
metrics.emHeightAscent
```

✓MDN

```
metrics.emHeightDescent
```

✓MDN

```
metrics.hangingBaseline
```

MDN

```
metrics.alphabeticBaseline
```

MDN

```
metrics.ideographicBaseline
```

MDN

Returns the measurement described below.

Objects that implement the [CanvasText](#) interface provide the following methods for rendering text.

The `fillText(text, x, y, maxWidth)` and `strokeText(text, x, y, maxWidth)` methods render the given *text* at the given (x, y) coordinates ensuring that the text isn't

wider than *maxWidth* if specified, using the current [font](#), [textAlign](#), and [textBaseline](#) values. Specifically, when the methods are invoked, the user agent must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. Run the [text preparation algorithm](#), passing it *text*, the object implementing the [CanvasText](#) interface, and, if the *maxWidth* argument was provided, that argument. Let *glyphs* be the result.
3. Move all the shapes in *glyphs* to the right by x [CSS pixels](#) and down by y [CSS pixels](#).
4. Paint the shapes given in *glyphs*, as transformed by the [current transformation matrix](#), with each [CSS pixel](#) in the coordinate space of *glyphs* mapped to one coordinate space unit.

For [fillText\(\)](#), [this](#)'s [fill style](#) must be applied to the shapes and [this](#)'s [stroke style](#) must be ignored. For [strokeText\(\)](#), the reverse holds: [this](#)'s [stroke style](#) must be applied to the result of [tracing](#) the shapes using the object implementing the [CanvasText](#) interface for the line styles, and [this](#)'s [fill style](#) must be ignored.

These shapes are painted without affecting the current path, and are subject to [shadow effects](#), [global alpha](#), the [clipping region](#), and the [current compositing and blending operator](#).

The [measureText\(text\)](#) method steps are to run the [text preparation algorithm](#), passing it *text* and the object implementing the [CanvasText](#) interface, and then using the returned [inline box](#) must return a new [TextMetrics](#) object with members behaving as described in the following list: [\[CSS\]](#)

#### **width attribute**

The width of that [inline box](#), in [CSS pixels](#). (The text's advance width.)

#### **actualBoundingBoxLeft attribute**

The distance parallel to the baseline from the alignment point given by the [textAlign](#) attribute to the left side of the bounding rectangle of the given text, in [CSS pixels](#); positive numbers indicating a distance going left from the given alignment point.

*The sum of this value and the next ([actualBoundingBoxRight](#)) can be wider than the width of the [inline box](#) ([width](#)), in particular with slanted fonts where characters overhang their advance width.*

#### **actualBoundingBoxRight attribute**

The distance parallel to the baseline from the alignment point given by the [`textAlign`](#) attribute to the right side of the bounding rectangle of the given text, in [`CSS pixels`](#); positive numbers indicating a distance going right from the given alignment point.

#### **fontBoundingBoxAscent attribute**

The distance from the horizontal line indicated by the [`textBaseline`](#) attribute to the [`ascent metric`](#) of the [`first available font`](#), in [`CSS pixels`](#); positive numbers indicating a distance going up from the given baseline.

*This value and the next are useful when rendering a background that have to have a consistent height even if the exact text being rendered changes. The [`actualBoundingBoxAscent`](#) attribute (and its corresponding attribute for the descent) are useful when drawing a bounding box around specific text.*

#### **fontBoundingBoxDescent attribute**

The distance from the horizontal line indicated by the [`textBaseline`](#) attribute to the [`descent metric`](#) of the [`first available font`](#), in [`CSS pixels`](#); positive numbers indicating a distance going down from the given baseline.

#### **actualBoundingBoxAscent attribute**

The distance from the horizontal line indicated by the [`textBaseline`](#) attribute to the top of the bounding rectangle of the given text, in [`CSS pixels`](#); positive numbers indicating a distance going up from the given baseline.

*This number can vary greatly based on the input text, even if the first font specified covers all the characters in the input. For example, the [`actualBoundingBoxAscent`](#) of a lowercase "o" from an [`alphabetic baseline`](#) would be less than that of an uppercase "F". The value can easily be negative; for example, the distance from the top of the em box ([`textBaseline`](#) value [`"top"`](#)) to the top of the bounding rectangle when the given text is just a single comma ", " would likely (unless the font is quite unusual) be negative.*

#### **actualBoundingBoxDescent attribute**

The distance from the horizontal line indicated by the [`textBaseline`](#) attribute to the bottom of the bounding rectangle of the given text, in [`CSS pixels`](#); positive numbers indicating a distance going down from the given baseline.

#### **emHeightAscent attribute**

The distance from the horizontal line indicated by the [`textBaseline`](#) attribute to the highest top of the em squares in the [`inline box`](#), in [`CSS pixels`](#); positive numbers indicating that the given baseline is below the top of that em square (so this value will usually be positive). Zero if the given baseline is the top of that em square; half the font size if the given baseline is the middle of that em square.

#### **emHeightDescent attribute**

The distance from the horizontal line indicated by the [textBaseline](#) attribute to the lowest bottom of the em squares in the [inline box](#), in [CSS pixels](#); positive numbers indicating that the given baseline is above the bottom of that em square. (Zero if the given baseline is the bottom of that em square.)

#### **hangingBaseline attribute**

The distance from the horizontal line indicated by the [textBaseline](#) attribute to the [hanging baseline](#) of the [inline box](#), in [CSS pixels](#); positive numbers indicating that the given baseline is below the [hanging baseline](#). (Zero if the given baseline is the [hanging baseline](#).)

#### **alphabeticBaseline attribute**

The distance from the horizontal line indicated by the [textBaseline](#) attribute to the [alphabetic baseline](#) of the [inline box](#), in [CSS pixels](#); positive numbers indicating that the given baseline is below the [alphabetic baseline](#). (Zero if the given baseline is the [alphabetic baseline](#).)

#### **ideographicBaseline attribute**

The distance from the horizontal line indicated by the [textBaseline](#) attribute to the [ideographic-under baseline](#) of the [inline box](#), in [CSS pixels](#); positive numbers indicating that the given baseline is below the [ideographic-under baseline](#). (Zero if the given baseline is the [ideographic-under baseline](#).)

*Glyphs rendered using [fillText\(\)](#) and [strokeText\(\)](#) can spill out of the box given by the font size (the em square size) and the width returned by [measureText\(\)](#) (the text width). Authors are encouraged to use the bounding box values described above if this is an issue.*

*A future version of the 2D context API might provide a way to render fragments of documents, rendered using CSS, straight to the canvas. This would be provided in preference to a dedicated way of doing multiline layout.*

#### **4.12.5.1.12 Drawing paths to the canvas**

Objects that implement the [CanvasDrawPath](#) interface have a **current default path**. There is only one [current default path](#), it is not part of the [drawing state](#). The [current default path](#) is a [path](#), as described above.

```
context.beginPath()
```



Resets the [current default path](#).

```
context.fill([ fillRule ])
```





```
context.fill(path [, fillRule ])
```

Fills the subpaths of the [current default path](#) or the given path with the current fill style, obeying the given fill rule.

```
context.stroke()
```

✓MDN

```
context.stroke(path)
```

Strokes the subpaths of the [current default path](#) or the given path with the current stroke style.

```
context.clip([ fillRule ])
```

✓MDN

```
context.clip(path [, fillRule ])
```

Further constrains the clipping region to the [current default path](#) or the given path, using the given fill rule to determine what points are in the path.

```
context.isPointInPath(x, y [, fillRule ])
```

✓MDN

```
context.isPointInPath(path, x, y [, fillRule ])
```

Returns true if the given point is in the [current default path](#) or the given path, using the given fill rule to determine what points are in the path.

```
context.isPointInStroke(x, y)
```

✓MDN

```
context.isPointInStroke(path, x, y)
```

Returns true if the given point would be in the region covered by the stroke of the [current default path](#) or the given path, given the current stroke style.

The `beginPath()` method steps are to empty the list of subpaths in [this](#)'s [current default path](#) so that it once again has zero subpaths.

Where the following method definitions use the term **intended path** for a [Path2D](#)-or-null *path*, it means *path* itself if it is a [Path2D](#) object, or the [current default path](#) otherwise.

When the [intended path](#) is a [Path2D](#) object, the coordinates and lines of its subpaths must be transformed according to the [current transformation matrix](#) on the object implementing the [CanvasTransform](#) interface when used by these methods (without affecting the [Path2D](#) object itself). When the intended path is the [current default path](#), it is not affected by the transform. (This is because transformations already affect the [current default path](#) when it is constructed, so applying it when it is painted as well would result in a double transformation.)

The `fill(fillRule)` method steps are to run the [fill steps](#) given [this](#), null, and *fillRule*.

The `fill(path, fillRule)` method steps are to run the [fill steps](#) given [this](#), `path`, and `fillRule`.

The **fill steps**, given a [CanvasDrawPath](#) `context`, a [Path2D](#)-or-null `path`, and a [fill rule](#) `fillRule`, are to fill all the subpaths of the [intended path](#) for `path`, using `context`'s [fill style](#), and using the [fill rule](#) indicated by `fillRule`. Open subpaths must be implicitly closed when being filled (without affecting the actual subpaths).

The `stroke()` method steps are to run the [stroke steps](#) given [this](#) and null.

The `stroke(path)` method steps are to run the [stroke steps](#) given [this](#) and `path`.

The **stroke steps**, given a [CanvasDrawPath](#) `context` and a [Path2D](#)-or-null `path`, are to [trace](#) the [intended path](#) for `path`, using `context`'s line styles as set by its [CanvasPathDrawingStyles](#) mixin, and then fill the resulting path using `context`'s [stroke style](#), using the [nonzero winding rule](#).

*As a result of how the algorithm to [trace a path](#) is defined, overlapping parts of the paths in one stroke operation are treated as if their union was what was painted. The stroke style is affected by the transformation during painting, even if the [current default path](#) is used.*

Paths, when filled or stroked, must be painted without affecting the [current default path](#) or any [Path2D](#) objects, and must be subject to [shadow effects](#), [global alpha](#), the [clipping region](#), and the [current compositing and blending operator](#). (The effect of transformations is described above and varies based on which path is being used.)

---

The `clip(fillRule)` method steps are to run the [clip steps](#) given [this](#), null, and `fillRule`.

The `clip(path, fillRule)` method steps are to run the [clip steps](#) given [this](#), `path`, and `fillRule`.

The **clip steps**, given a [CanvasDrawPath](#) `context`, a [Path2D](#)-or-null `path`, and a [fill rule](#) `fillRule`, are to create a new **clipping region** by calculating the intersection of `context`'s current clipping region and the area described by the [intended path](#) for `path`, using the [fill rule](#) indicated by `fillRule`. Open subpaths must be implicitly closed when computing the clipping region, without affecting the actual subpaths. The new clipping region replaces the current clipping region.

When the context is initialized, its current clipping region must be set to the largest infinite surface (i.e. by default, no clipping occurs).

---

The `isPointInPath(x, y, fillRule)` method steps are to return the result of the [is point in path steps](#) given [this](#), null, x, y, and *fillRule*.

The `isPointInPath(path, x, y, fillRule)` method steps are to return the result of the [is point in path steps](#) given [this](#), null, x, y, and *fillRule*.

The **is point in path steps**, given a [CanvasDrawPath](#) *context*, a [Path2D](#)-or-null *path*, two numbers x and y, and a [fill rule](#) *fillRule*, are:

1. If x or y are infinite or NaN, then return false.
2. If the point given by the x and y coordinates, when treated as coordinates in the canvas coordinate space unaffected by the current transformation, is inside the [intended path](#) for *path* as determined by the [fill rule](#) indicated by *fillRule*, then return true. Open subpaths must be implicitly closed when computing the area inside the path, without affecting the actual subpaths. Points on the path itself must be considered to be inside the path.
3. Return false.

---

The `isPointInStroke(x, y)` method steps are to return the result of the [is point in stroke steps](#) given [this](#), null, x, and y.

The `isPointInStroke(path, x, y)` method steps are to return the result of the [is point in stroke steps](#) given [this](#), *path*, x, and y.

The **is point in stroke steps**, given a [CanvasDrawPath](#) *context*, a [Path2D](#)-or-null *path*, and two numbers x and y, are:

1. If x or y are infinite or NaN, then return false.
2. If the point given by the x and y coordinates, when treated as coordinates in the canvas coordinate space unaffected by the current transformation, is inside the path that results from [tracing](#) the [intended path](#) for *path*, using the [nonzero winding rule](#), and using *context*'s line styles as set by its [CanvasPathDrawingStyles](#) mixin, then return true. Points on the resulting path must be considered to be inside the path.
3. Return false.

---

This canvas element has a couple of checkboxes. The path-related commands are highlighted:

```
<canvas height=400 width=750>
  <label><input type=checkbox id=showA> Show As</label>
  <label><input type=checkbox id=showB> Show Bs</label>
  <!-- ... -->
</canvas>
<script>
  function drawCheckbox(context, element, x, y, paint) {
    context.save();
    context.font = '10px sans-serif';
    context.textAlign = 'left';
    context.textBaseline = 'middle';
    var metrics =
context.measureText(element.labels[0].textContent);
    if (paint) {
      context.beginPath();
      context.strokeStyle = 'black';
      context.rect(x-5, y-5, 10, 10);
      context.stroke();
      if (element.checked) {
        context.fillStyle = 'black';
        context.fill();
      }
      context.fillText(element.labels[0].textContent, x+5, y);
    }
    context.beginPath();
    context.rect(x-7, y-7, 12 + metrics.width+2, 14);

    context.drawFocusIfNeeded(element);
    context.restore();
  }
  function drawBase() { /* ... */ }
  function drawAs() { /* ... */ }
  function drawBs() { /* ... */ }
  function redraw() {
    var canvas = document.getElementsByTagName('canvas')[0];
```

```

    var context = canvas.getContext('2d');
    context.clearRect(0, 0, canvas.width, canvas.height);
    drawCheckbox(context, document.getElementById('showA'), 20, 40,
true);
    drawCheckbox(context, document.getElementById('showB'), 20, 60,
true);
    drawBase();
    if (document.getElementById('showA').checked)
        drawAs();
    if (document.getElementById('showB').checked)
        drawBs();
}
function processClick(event) {
    var canvas = document.getElementsByTagName('canvas')[0];
    var context = canvas.getContext('2d');
    var x = event.clientX;
    var y = event.clientY;
    var node = event.target;
    while (node) {
        x -= node.offsetLeft - node.scrollLeft;
        y -= node.offsetTop - node.scrollTop;
        node = node.offsetParent;
    }
    drawCheckbox(context, document.getElementById('showA'), 20, 40,
false);
    if (context.isPointInPath(x, y))
        document.getElementById('showA').checked =
!(document.getElementById('showA').checked);
    drawCheckbox(context, document.getElementById('showB'), 20, 60,
false);
    if (context.isPointInPath(x, y))
        document.getElementById('showB').checked =
!(document.getElementById('showB').checked);
    redraw();
}

document.getElementsByTagName('canvas')[0].addEventListener('focus'
, redraw, true);

document.getElementsByTagName('canvas')[0].addEventListener('blur',
redraw, true);

document.getElementsByTagName('canvas')[0].addEventListener('change
', redraw, true);

```

```
document.getElementsByTagName('canvas')[0].addEventListener('click', processClick, false);

redraw();

</script>
```

#### 4.12.5.1.13 Drawing focus rings and scrolling paths into view

`context.drawFocusIfNeeded(element)`



`context.drawFocusIfNeeded(path, element)`

Si l'élément donné est [focus](#), dessine un anneau de focus autour du [chemin par défaut actuel](#) ou du chemin donné, en suivant les conventions de la plateforme pour les anneaux de focus.

`context.scrollPathIntoView()`



`context.scrollPathIntoView(path)`

Fait défiler le [chemin par défaut actuel](#) ou le chemin donné dans la vue. Ceci est particulièrement utile sur les appareils dotés de petits écrans, où l'ensemble de la toile peut ne pas être visible en même temps.

Les objets qui implémentent l'[CanvasUserInterface](#) interface fournissent les méthodes suivantes pour contrôler le dessin des cercles de focus et le défilement des chemins dans la vue.

---

La méthode, lorsqu'elle est invoquée, doit exécuter ces étapes : `drawFocusIfNeeded(element)`

1. Si l'élément n'est pas [focalisé](#) ou n'est pas un descendant de l'élément avec le contexte duquel la méthode est associée, alors retourne.
2. Draw a focus ring of the appropriate style along the intended path, following platform conventions.

*Some platforms only draw focus rings around elements that have been focused from the keyboard, and not those focused from the mouse. Other platforms simply don't draw focus rings around some elements at all unless relevant accessibility features are enabled. This API is intended to follow these conventions. User agents that implement distinctions based on the manner in which the element was focused are encouraged to classify focus driven by*

*the `focus()` method based on the kind of user interaction event from which the call was triggered (if any).*

The focus ring should not be subject to the [shadow effects](#), the [global alpha](#), the [current compositing and blending operator](#), the [fill style](#), the [stroke style](#), or any of the members in the [CanvasPathDrawingStyles](#), [CanvasTextDrawingStyles](#) interfaces, but *should* be subject to the [clipping region](#). (The effect of transformations is described above and varies based on which path is being used.)

3. [Inform the user](#) that the focus is at the location given by the intended path. User agents may wait until the next time the [event loop](#) reaches its [update the rendering](#) step to optionally inform the user.

User agents should not implicitly close open subpaths in the intended path when drawing the focus ring.

*This might be a moot point, however. For example, if the focus ring is drawn as an axis-aligned bounding rectangle around the points in the intended path, then whether the subpaths are closed or not has no effect. This specification intentionally does not specify precisely how focus rings are to be drawn: user agents are expected to honor their platform's native conventions.*

---

The `scrollPathIntoView()` method, when invoked, must run these steps:

1. Let *specifiedRectangle* be the rectangle of the bounding box of the intended path.
2. Let *notionalChild* be a hypothetical element that is a rendered child of the [canvas](#) element whose dimensions are those of *specifiedRectangle*.
3. [Scroll \*notionalChild\* into view](#) with *behavior* set to "auto", *block* set to "start", and *inline* set to "nearest".
4. Optionally, [inform the user](#) that the caret or selection (or both) cover *specifiedRectangle* of the canvas. The user agent may wait until the next time the [event loop](#) reaches its [update the rendering](#) step to optionally inform the user.

"Inform the user", as used in this section, does not imply any persistent state change. It could mean, for instance, calling a system accessibility API to notify assistive technologies such as magnification tools so that the user's magnifier moves to the given area of the canvas. However, it does not associate the path with the element, or provide a region for tactile feedback, etc.

#### 4.12.5.1.14 Drawing images

Objects that implement the [CanvasDrawImage](#) interface have the `drawImage()` method to draw images.

This method can be invoked with three different sets of arguments:

- `drawImage(image, dx, dy)`
- `drawImage(image, dx, dy, dw, dh)`
- `drawImage(image, sx, sy, sw, sh, dx, dy, dw, dh)`

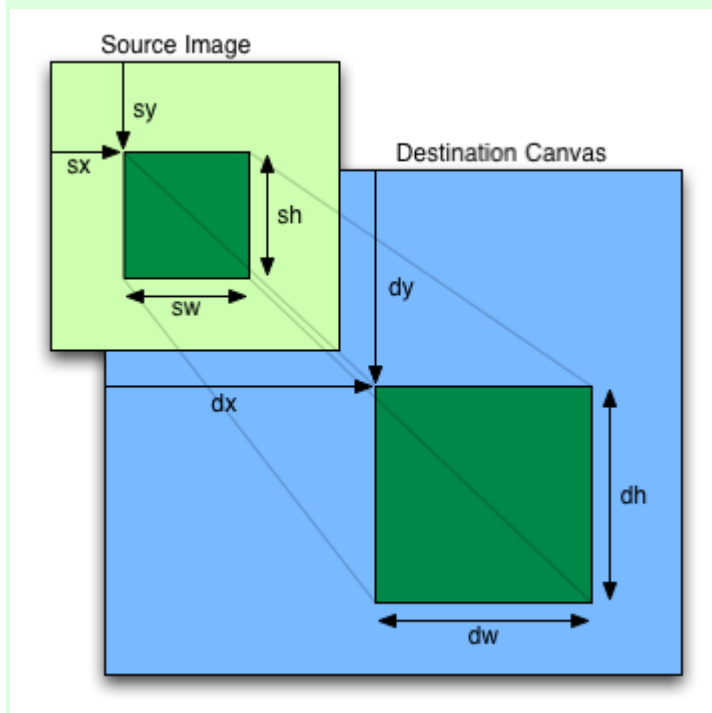
`context.drawImage(image, dx, dy)`

✓MDN

`context.drawImage(image, dx, dy, dw, dh)`

`context.drawImage(image, sx, sy, sw, sh, dx, dy, dw, dh)`

Draws the given image onto the canvas. The arguments are interpreted as follows:



If the image isn't yet fully decoded, then nothing is drawn. If the image is a canvas with no data, throws an `"InvalidStateError"` [DOMException](#).

When the `drawImage()` method is invoked, the user agent must run these steps:

1. If any of the arguments are infinite or NaN, then return.
2. Let *usability* be the result of [checking the usability of image](#).
3. If *usability* is *bad*, then return (without drawing anything).
4. Establish the source and destination rectangles as follows:



If not specified, the *dw* and *dh* arguments must default to the values of *sw* and *sh*, interpreted such that one [CSS pixel](#) in the image is treated as one unit in the [output bitmap](#)'s coordinate space. If the *sx*, *sy*, *sw*, and *sh* arguments are omitted, then they must default to 0, 0, the image's [intrinsic width](#) in image pixels, and the image's [intrinsic height](#) in image pixels, respectively. If the image has no [intrinsic dimensions](#), then the *concrete object* size must be used instead, as determined using the CSS "[Concrete Object Size Resolution](#)" algorithm, with the *specified size* having neither a definite width nor height, nor any additional constraints, the object's intrinsic properties being those of the *image* argument, and the [default object size](#) being the size of the [output bitmap](#). [\[CSSIMAGES\]](#)

The source rectangle is the rectangle whose corners are the four points (*sx*, *sy*), (*sx*+*sw*, *sy*), (*sx*+*sw*, *sy*+*sh*), (*sx*, *sy*+*sh*).

The destination rectangle is the rectangle whose corners are the four points (*dx*, *dy*), (*dx*+*dw*, *dy*), (*dx*+*dw*, *dy*+*dh*), (*dx*, *dy*+*dh*).

When the source rectangle is outside the source image, the source rectangle must be clipped to the source image and the destination rectangle must be clipped in the same proportion.

*When the destination rectangle is outside the destination image (the [output bitmap](#)), the pixels that land outside the [output bitmap](#) are discarded, as if the destination was an infinite canvas whose rendering was clipped to the dimensions of the [output bitmap](#).*

5. If one of the *sw* or *sh* arguments is zero, then return. Nothing is painted.
6. Paint the region of the *image* argument specified by the source rectangle on the region of the rendering context's [output bitmap](#) specified by the destination rectangle, after applying the [current transformation matrix](#) to the destination rectangle.

The image data must be processed in the original direction, even if the dimensions given are negative.

When scaling up, if the [imageSmoothingEnabled](#) attribute is set to true, the user agent should attempt to apply a smoothing algorithm to the image data when it is scaled. User agents which support multiple filtering algorithms may use the value of the [imageSmoothingQuality](#) attribute to guide the choice of filtering algorithm when the [imageSmoothingEnabled](#) attribute is set to true. Otherwise, the image must be rendered using nearest-neighbor interpolation.

*This specification does not define the precise algorithm to use when scaling an image down, or when scaling an image up when the [imageSmoothingEnabled](#) attribute is set to true.*  
*When a [canvas](#) element is drawn onto itself, the [drawing model](#) requires the source to be copied before the image is drawn, so it is possible to copy parts of a [canvas](#) element onto overlapping parts of itself.*

If the original image data is a bitmap image, then the value painted at a point in the destination rectangle is computed by filtering the original image data. The user agent may use any filtering algorithm (for example bilinear interpolation or nearest-neighbor). When the filtering algorithm requires a pixel value from outside the original image data, it must instead use the value from the nearest edge pixel. (That is, the filter uses 'clamp-to-edge' behavior.) When the filtering algorithm requires a pixel value from outside the source rectangle but inside the original image data, then the value from the original image data must be used.

*Thus, scaling an image in parts or in whole will have the same effect. This does mean that when sprites coming from a single sprite sheet are to be scaled, adjacent images in the sprite sheet can interfere. This can be avoided by ensuring each sprite in the sheet is surrounded by a border of [transparent black](#), or by copying sprites to be scaled into temporary [canvas](#) elements and drawing the scaled sprites from there.*

Images are painted without affecting the current path, and are subject to [shadow effects](#), [global alpha](#), the [clipping region](#), and the [current compositing and blending operator](#).

7. If *image* is [not origin-clean](#), then set the [CanvasRenderingContext2D](#)'s [origin-clean](#) flag to false.

#### 4.12.5.1.15 Pixel manipulation

```
imagedata = new ImageData(sw, sh [, settings])
```

✓MDN

Returns an [ImageData](#) object with the given dimensions and the color space indicated by *settings*. All the pixels in the returned object are [transparent black](#).

Throws an "[IndexSizeError](#)" [DOMException](#) if either of the width or height arguments are zero.

```
imagedata = new ImageData(data, sw [, sh [, settings ] ])
```

Returns an [ImageData](#) object using the data provided in the [Uint8ClampedArray](#) argument, interpreted using the given dimensions and the color space indicated by *settings*.

As each pixel in the data is represented by four numbers, the length of the data needs to be a multiple of four times the given width. If the height is provided as well, then the length needs to be exactly the width times the height times 4.

Throws an "[IndexSizeError](#)" [DOMException](#) if the given data and dimensions can't be interpreted consistently, or if either dimension is zero.

```
imagedata = context.createImageData(imagedata)
```

Returns an [ImageData](#) object with the same dimensions and color space as the argument. All the pixels in the returned object are [transparent black](#).

```
imagedata = context.createImageData(sw, sh [, settings])
```

✓MDN

Returns an [ImageData](#) object with the given dimensions. The color space of the returned object is the [color space](#) of *context* unless overridden by *settings*. All the pixels in the returned object are [transparent black](#).

Throws an "[IndexSizeError](#)" [DOMException](#) if either of the width or height arguments are zero.

```
imagedata = context.getImageData(sx, sy, sw, sh [, settings])
```

✓MDN

Returns an [ImageData](#) object containing the image data for the given rectangle of the bitmap. The color space of the returned object is the [color space](#) of *context* unless overridden by *settings*.

Throws an "[IndexSizeError](#)" [DOMException](#) if the either of the width or height arguments are zero.

```
imagedata.width
```

✓MDN

```
imagedata.height
```

✓MDN

Returns the actual dimensions of the data in the [ImageData](#) object, in pixels.

```
imagedata.data
```

✓MDN

Returns the one-dimensional array containing the data in RGBA order, as integers in the range 0 to 255.

```
imagedata.colorSpace
```

Returns the color space of the pixels.

```
context.putImageData(imagedata, dx, dy [, dirtyX, dirtyY, dirtyWidth, dirtyHeight ])
```

✓MDN

Paints the data from the given [ImageData](#) object onto the bitmap. If a dirty rectangle is provided, only the pixels from that rectangle are painted.

The [globalAlpha](#) and [globalCompositeOperation](#) properties, as well as the [shadow attributes](#), are ignored for the purposes of this method call; pixels in the canvas are replaced wholesale, with no composition, alpha blending, no shadows, etc.

Throws an `"InvalidStateError"` `DOMException` if the `imagedata` object's `data` attribute value's `[[ViewedArrayBuffer]]` internal slot is detached.

Objects that implement the `CanvasImageData` interface provide the following methods for reading and writing pixel data to the bitmap.

The `new ImageData(sw, sh, settings)` constructor steps are:

1. If one or both of `sw` and `sh` are zero, then throw an `"IndexSizeError"` `DOMException`.
2. [Initialize this](#) given `sw`, `sh`, and `settings` set to `settings`.
3. Initialize the image data of [this](#) to [transparent black](#).

The `new ImageData(data, sw, sh, settings)` constructor steps are:

1. Let `length` be the number of bytes in `data`.
2. If `length` is not a nonzero integral multiple of four, then throw an `"InvalidStateError"` `DOMException`.
3. Let `length` be `length` divided by four.
4. If `length` is not an integral multiple of `sw`, then throw an `"IndexSizeError"` `DOMException`.

*At this step, the `length` is guaranteed to be greater than zero (otherwise the second step above would have aborted the steps), so if `sw` is zero, this step will throw the exception and return.*

5. Let `height` be `length` divided by `sw`.
6. If `sh` was given and its value is not equal to `height`, then throw an `"IndexSizeError"` `DOMException`.
7. [Initialize this](#) given `sw`, `sh`, `settings` set to `settings`, and `source` set to `data`.

*This step does not set [this](#)'s data to a copy of `data`. It sets it to the actual `Uint8ClampedArray` object passed as `data`.*

The `createImageData(sw, sh, settings)` method steps are:

1. If one or both of `sw` and `sh` are zero, then throw an `"IndexSizeError"` `DOMException`.
2. Let `newImageData` be a [new](#) `ImageData` object.

3. [Initialize](#) *newImageData* given the absolute magnitude of *sw*, the absolute magnitude of *sh*, [settings](#) set to *settings*, and [defaultColorSpace](#) set to *this*'s [color space](#).
4. Initialize the image data of *newImageData* to [transparent black](#).
5. Return *newImageData*.

The [createImageData \(imagedata\)](#) method steps are:

1. Let *newImageData* be a [new ImageData](#) object.
2. [Initialize](#) *newImageData* given the value of *imagedata*'s [width](#) attribute, the value of *imagedata*'s [height](#) attribute, and [defaultColorSpace](#) set to the value of *imagedata*'s [colorSpace](#) attribute.
3. Initialize the image data of *newImageData* to [transparent black](#).
4. Return *newImageData*.

The [getImageData \(sx, sy, sw, sh, settings\)](#) method steps are:

1. If either the *sw* or *sh* arguments are zero, then throw an ["IndexSizeError" DOMException](#).
2. If the [CanvasRenderingContext2D](#)'s [origin-clean](#) flag is set to false, then throw a ["SecurityError" DOMException](#).
3. Let *imageData* be a [new ImageData](#) object.
4. [Initialize](#) *imageData* given *sw*, *sh*, [settings](#) set to *settings*, and [defaultColorSpace](#) set to *this*'s [color space](#).
5. Let the source rectangle be the rectangle whose corners are the four points (*sx*, *sy*), (*sx*+*sw*, *sy*), (*sx*+*sw*, *sy*+*sh*), (*sx*, *sy*+*sh*).
6. Set the pixel values of *imageData* to be the pixels of *this*'s [output bitmap](#) in the area specified by the source rectangle in the bitmap's coordinate space units, converted from *this*'s [color space](#) to *imageData*'s [colorSpace](#) using ['relative-colorimetric'](#) rendering intent.
7. Set the pixels values of *imageData* for areas of the source rectangle that are outside of the [output bitmap](#) to [transparent black](#).
8. Return *imageData*.

To **initialize an ImageData object** *imageData*, given a positive integer number of rows *rows*, a positive integer number of pixels per row *pixelsPerRow*, an optional [ImageDataSettings](#) **settings**, an

optional [Uint8ClampedArray](#) **source**, and an  
optional [PredefinedColorSpace](#) **defaultColorSpace**:

## MDN

1. If *source* was given, then initialize the **data** attribute of *imageData* to *source*.
2. Otherwise (*source* was not given), initialize the **data** attribute of *imageData* to a new [Uint8ClampedArray](#) object. The [Uint8ClampedArray](#) object must use a new [Canvas Pixel ArrayBuffer](#) for its storage, and must have a zero start offset and a length equal to the length of its storage, in bytes. The [Canvas Pixel ArrayBuffer](#) must have the correct size to store *rows* × *pixelsPerRow* pixels.

If the [Canvas Pixel ArrayBuffer](#) cannot be allocated, then rethrow the [RangeError](#) thrown by JavaScript, and return.

3. Initialize the **width** attribute of *imageData* to *pixelsPerRow*.
4. Initialize the **height** attribute of *imageData* to *rows*.
5. If *settings* was given and *settings*["[colorSpace](#)"] [exists](#), then initialize the **colorSpace** attribute of *imageData* to *settings*["[colorSpace](#)"].
6. Otherwise, if *defaultColorSpace* was given, then initialize the **colorSpace** attribute of *imageData* to *defaultColorSpace*.
7. Otherwise, initialize the **colorSpace** attribute of *imageData* to "[srgb](#)".

[ImageData](#) objects are [serializable objects](#). Their [serialization steps](#), given *value* and *serialized*, are:

1. Set *serialized*.[[Data]] to the [sub-serialization](#) of the value of *value*'s **data** attribute.
2. Set *serialized*.[[Width]] to the value of *value*'s **width** attribute.
3. Set *serialized*.[[Height]] to the value of *value*'s **height** attribute.
4. Set *serialized*.[[ColorSpace]] to the value of *value*'s **colorSpace** attribute.

Their [deserialization steps](#), given *serialized*, *value*, and *targetRealm*, are:

1. Initialize *value*'s **data** attribute to the [sub-deserialization](#) of *serialized*.[[Data]].
2. Initialize *value*'s **width** attribute to *serialized*.[[Width]].
3. Initialize *value*'s **height** attribute to *serialized*.[[Height]].

4. Initialize *value*'s colorSpace attribute to *serialized*.[[ColorSpace]].

A **Canvas Pixel ArrayBuffer** is an ArrayBuffer whose data is represented in left-to-right order, row by row top to bottom, starting with the top left, with each pixel's red, green, blue, and alpha components being given in that order for each pixel. Each component of each pixel represented in this array must be in the range 0..255, representing the 8 bit value for that component. The components must be assigned consecutive indices starting with 0 for the top left pixel's red component.

The **putImageData()** method writes data from ImageData structures back to the rendering context's output bitmap. Its arguments are: *imagedata*, *dx*, *dy*, *dirtyX*, *dirtyY*, *dirtyWidth*, and *dirtyHeight*.

When the last four arguments to this method are omitted, they must be assumed to have the values 0, 0, the width member of the *imagedata* structure, and the height member of the *imagedata* structure, respectively.

The method, when invoked, must act as follows:

1. Let *buffer* be *imagedata*'s data attribute value's [[ViewedArrayBuffer]] internal slot.
2. If IsDetachedBuffer(*buffer*) is true, then throw an "InvalidStateError" DOMException.
3. If *dirtyWidth* is negative, then let *dirtyX* be *dirtyX*+*dirtyWidth*, and let *dirtyWidth* be equal to the absolute magnitude of *dirtyWidth*.  
  
If *dirtyHeight* is negative, then let *dirtyY* be *dirtyY*+*dirtyHeight*, and let *dirtyHeight* be equal to the absolute magnitude of *dirtyHeight*.
4. If *dirtyX* is negative, then let *dirtyWidth* be *dirtyWidth*+*dirtyX*, and let *dirtyX* be zero.  
  
If *dirtyY* is negative, then let *dirtyHeight* be *dirtyHeight*+*dirtyY*, and let *dirtyY* be zero.
5. If *dirtyX*+*dirtyWidth* is greater than the width attribute of the *imagedata* argument, then let *dirtyWidth* be the value of that width attribute, minus the value of *dirtyX*.  
  
If *dirtyY*+*dirtyHeight* is greater than the height attribute of the *imagedata* argument, then let *dirtyHeight* be the value of that height attribute, minus the value of *dirtyY*.
6. If, after those changes, either *dirtyWidth* or *dirtyHeight* are negative or zero, then return without affecting any bitmaps.
7. For all integer values  
of *x* and *y* where  $dirtyX \leq x < dirtyX + dirtyWidth$  and  $dirtyY \leq y < dirtyY + dirtyHeight$



ght, copy the four channels of the pixel with coordinate (x, y) in the *imagedata* data structure's [Canvas Pixel](#) [ArrayBuffer](#) to the pixel with coordinate (dx+x, dy+y) in the rendering context's [output bitmap](#).

*Due to the lossy nature of converting between color spaces and converting to and from [premultiplied alpha](#) color values, pixels that have just been set using [putImageData\(\)](#), and are not completely opaque, might be returned to an equivalent [getImageData\(\)](#) as different values.*

The current path, [transformation matrix](#), [shadow attributes](#), [global alpha](#), the [clipping region](#), and [current compositing and blending operator](#) must not affect the methods described in this section.

In the following example, the script generates an [ImageData](#) object so that it can draw onto it.

```
// canvas is a reference to a <canvas> element
var context = canvas.getContext('2d');

// create a blank slate
var data = context.createImageData(canvas.width, canvas.height);

// create some plasma
FillPlasma(data, 'green'); // green plasma

// add a cloud to the plasma
AddCloud(data, data.width/2, data.height/2); // put a cloud in the
middle

// paint the plasma+cloud on the canvas
context.putImageData(data, 0, 0);

// support methods
function FillPlasma(data, color) { ... }
function AddCloud(data, x, y) { ... }
```

Here is an example of using [getImageData\(\)](#) and [putImageData\(\)](#) to implement an edge detection filter.

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>Edge detection demo</title>
    <script>
      var image = new Image();
```



```

function init() {
    image.onload = demo;
    image.src = "image.jpeg";
}

function demo() {
    var canvas = document.getElementsByTagName('canvas')[0];
    var context = canvas.getContext('2d');

    // draw the image onto the canvas
    context.drawImage(image, 0, 0);

    // get the image data to manipulate
    var input = context.getImageData(0, 0, canvas.width,
    canvas.height);

    // get an empty slate to put the data into
    var output = context.createImageData(canvas.width,
    canvas.height);

    // alias some variables for convenience
    // In this case input.width and input.height
    // match canvas.width and canvas.height
    // but we'll use the former to keep the code generic.
    var w = input.width, h = input.height;
    var inputData = input.data;
    var outputData = output.data;

    // edge detection
    for (var y = 1; y < h-1; y += 1) {
        for (var x = 1; x < w-1; x += 1) {
            for (var c = 0; c < 3; c += 1) {
                var i = (y*w + x)*4 + c;
                outputData[i] = 127 + -inputData[i - w*4 - 4] -
inputData[i - w*4] - inputData[i - w*4 + 4] +
                                -inputData[i - 4] +
8*inputData[i] - inputData[i + 4] +
                                -inputData[i + w*4 - 4] -
inputData[i + w*4] - inputData[i + w*4 + 4];
            }
            outputData[(y*w + x)*4 + 3] = 255; // alpha
        }
    }
}

```

```

    // put the image data back after manipulation
    context.putImageData(output, 0, 0);
  }
</script>
</head>
<body onload="init()">
  <canvas></canvas>
</body>
</html>

```

Here is an example of color space conversion applied when drawing a solid color and reading the result back using [`getImageData\(\)`](#).

```

<!DOCTYPE HTML>
<html lang="en">
<title>Color space image data demo</title>

<canvas></canvas>

<script>
const canvas = document.querySelector('canvas');
const context = canvas.getContext('2d', {colorSpace:'display-p3'});

// Draw a red rectangle. Note that the hex color notation
// specifies sRGB colors.
context.fillStyle = "#FF0000";
context.fillRect(0, 0, 64, 64);

// Get the image data.
const pixels = context.getImageData(0, 0, 1, 1);

// This will print 'display-p3', reflecting the default behavior
// of returning image data in the canvas's color space.
console.log(pixels.colorSpace);

// This will print the values 234, 51, and 35, reflecting the
// red fill color, converted to 'display-p3'.
console.log(pixels.data[0]);
console.log(pixels.data[1]);
console.log(pixels.data[2]);
</script>

```

#### 4.12.5.1.16 Compositing

```
context.globalAlpha [ = value ]
```

✓MDN

Returns the current [global alpha](#) value applied to rendering operations.

Can be set, to change the [global alpha](#) value. Values outside of the range 0.0 .. 1.0 are ignored.

```
context.globalCompositeOperation [ = value ]
```

✓MDN

Returns the [current compositing and blending operator](#), from the values defined in *Compositing and Blending*. [\[COMPOSITE\]](#)

Can be set, to change the [current compositing and blending operator](#). Unknown values are ignored.

Objects that implement the [CanvasCompositing](#) interface have a [global alpha](#) value and a [current compositing and blending operator](#) value that both affect all the drawing operations on this object.

The **global alpha** value gives an alpha value that is applied to shapes and images before they are composited onto the [output bitmap](#). The value ranges from 0.0 (fully transparent) to 1.0 (no additional transparency). It must initially have the value 1.0.

The `globalAlpha` getter steps are to return [this](#)'s [global alpha](#).

The `globalAlpha` setter steps are:

1. If the given value is either infinite, NaN, or not in the range 0.0 to 1.0, then return.
2. Otherwise, set [this](#)'s [global alpha](#) to the given value.

The **current compositing and blending operator** value controls how shapes and images are drawn onto the [output bitmap](#), once they have had the [global alpha](#) and the [current transformation matrix](#) applied. Initially, it must be set to "[source-over](#)".

The `globalCompositeOperation` getter steps are to return [this](#)'s [current compositing and blending operator](#).

The `globalCompositeOperation` setter steps are:

1. If the given value is not [identical to](#) any of the values that the [<blend-mode>](#) or the [<composite-mode>](#) properties are defined to take, then return. [\[COMPOSITE\]](#)
2. Otherwise, set [this](#)'s [current compositing and blending operator](#) to the given value.

#### 4.12.5.1.17 Image smoothing

```
context.imageSmoothingEnabled [ = value ]
```

✓MDN

Returns whether pattern fills and the [drawImage\(\)](#) method will attempt to smooth images if their pixels don't line up exactly with the display, when scaling images up.

Can be set, to change whether images are smoothed (true) or not (false).

```
context.imageSmoothingQuality [ = value ]
```

MDN

Returns the current image-smoothing-quality preference.

Can be set, to change the preferred quality of image smoothing. The possible values are "[low](#)", "[medium](#)" and "[high](#)". Unknown values are ignored.

Objects that implement the [CanvasImageSmoothing](#) interface have attributes that control how image smoothing is performed.

The `imageSmoothingEnabled` attribute, on getting, must return the last value it was set to. On setting, it must be set to the new value. When the object implementing the [CanvasImageSmoothing](#) interface is created, the attribute must be set to true.

The `imageSmoothingQuality` attribute, on getting, must return the last value it was set to. On setting, it must be set to the new value. When the object implementing the [CanvasImageSmoothing](#) interface is created, the attribute must be set to "[low](#)".

#### 4.12.5.1.18 Shadows

All drawing operations on an object which implements the [CanvasShadowStyles](#) interface are affected by the four global shadow attributes.

```
context.shadowColor [ = value ]
```

✓MDN

Returns the current shadow color.

Can be set, to change the shadow color. Values that cannot be parsed as CSS colors are ignored.

```
context.shadowOffsetX [ = value ]
```

✓MDN

```
context.shadowOffsetY [ = value ]
```



Returns the current shadow offset.

Can be set, to change the shadow offset. Values that are not finite numbers are ignored.

```
context.shadowBlur [ = value ]
```



Returns the current level of blur applied to shadows.

Can be set, to change the blur level. Values that are not finite numbers greater than or equal to zero are ignored.

The **shadowColor** attribute sets the color of the shadow.

When the context is created, the **shadowColor** attribute initially must be [transparent black](#).

On getting, the [serialization of the color](#) must be returned.

On setting, the new value must be [parsed](#) with this **canvas** element and the color assigned. If parsing the value results in failure then it must be ignored, and the attribute must retain its previous value. [\[CSSCOLOR\]](#)

The **shadowOffsetX** and **shadowOffsetY** attributes specify the distance that the shadow will be offset in the positive horizontal and positive vertical distance respectively. Their values are in coordinate space units. They are not affected by the current transformation matrix.

When the context is created, the shadow offset attributes must initially have the value 0.

On getting, they must return their current value. On setting, the attribute being set must be set to the new value, except if the value is infinite or NaN, in which case the new value must be ignored.

The **shadowBlur** attribute specifies the level of the blurring effect. (The units do not map to coordinate space units, and are not affected by the current transformation matrix.)

When the context is created, the **shadowBlur** attribute must initially have the value 0.

On getting, the attribute must return its current value. On setting the attribute must be set to the new value, except if the value is negative, infinite or NaN, in which case the new value must be ignored.

**Shadows are only drawn if** the opacity component of the alpha component of the color of `shadowColor` is nonzero and either the `shadowBlur` is nonzero, or the `shadowOffsetX` is nonzero, or the `shadowOffsetY` is nonzero.

When shadows are drawn, they must be rendered as follows:

1. Let  $A$  be an infinite transparent black bitmap on which the source image for which a shadow is being created has been rendered.
2. Let  $B$  be an infinite transparent black bitmap, with a coordinate space and an origin identical to  $A$ .
3. Copy the alpha channel of  $A$  to  $B$ , offset by `shadowOffsetX` in the positive x direction, and `shadowOffsetY` in the positive y direction.
4. If `shadowBlur` is greater than 0:
  1. Let  $\sigma$  be half the value of `shadowBlur`.
  2. Perform a 2D Gaussian Blur on  $B$ , using  $\sigma$  as the standard deviation.User agents may limit values of  $\sigma$  to an implementation-specific maximum value to avoid exceeding hardware limitations during the Gaussian blur operation.
5. Set the red, green, and blue components of every pixel in  $B$  to the red, green, and blue components (respectively) of the color of `shadowColor`.
6. Multiply the alpha component of every pixel in  $B$  by the alpha component of the color of `shadowColor`.
7. The shadow is in the bitmap  $B$ , and is rendered as part of the drawing model described below.

If the current compositing and blending operator is "`copy`", then shadows effectively won't render (since the shape will overwrite the shadow).

#### 4.12.5.1.19 Filters

All drawing operations on an object which implements the `CanvasFilters` interface are affected by the global `filter` attribute.

```
context.filter [ = value ]
```

MDN

Returns the current filter.

Can be set, to change the filter. Values can either be the string "none" or a string parseable as a [<filter-value-list>](#). Other values are ignored.

Such objects have an associated **current filter**, which is a string. Initially the [current filter](#) is set to the string "none". Whenever the value of the [current filter](#) is the string "none" filters will be disabled for the context.

The [filter](#) getter steps are to return [this's current filter](#).

The [filter](#) setter steps are:

1. If the given value is "none", then set [this's current filter](#) to "none" and return.
2. Let *parsedValue* be the result of [parsing](#) the given values as a [<filter-value-list>](#). If any property-independent style sheet syntax like 'inherit' or 'initial' is present, then this parsing must return failure.
3. If *parsedValue* is failure, then return.
4. Set [this's current filter](#) to the given value.

*Though `context.filter = "none"` will disable filters for the context, `context.filter = ""`, `context.filter = null`, and `context.filter = undefined` are all treated as unparseable inputs and the value of the [current filter](#) is left unchanged.*

Coordinates used in the value of the [current filter](#) are interpreted such that one pixel is equivalent to one SVG user space unit and to one canvas coordinate space unit. Filter coordinates are not affected by the [current transformation matrix](#). The current transformation matrix affects only the input to the filter. Filters are applied in the [output bitmap's](#) coordinate space.

When the value of the [current filter](#) is a string parseable as a [<filter-value-list>](#) which defines lengths using percentages or using 'em' or 'ex' units, these must be interpreted relative to the [computed value](#) of the 'font-size' property of the [font style source object](#) at the time that the attribute is set. If the [computed values](#) are undefined for a particular case (e.g. because the [font style source object](#) is not an element or is not [being rendered](#)), then the relative keywords must be interpreted relative to the default value of the [font](#) attribute. The 'larger' and 'smaller' keywords are not supported.

If the value of the [current filter](#) is a string parseable as a [<filter-value-list>](#) with a reference to an SVG filter in the same document, and this SVG filter changes, then the changed filter is used for the next draw operation.

If the value of the [current filter](#) is a string parseable as a [<filter-value-list>](#) with a reference to an SVG filter in an external resource document and that document is not loaded when a drawing operation is invoked, then the drawing operation must proceed with no filtering.

#### 4.12.5.1.20 Working with externally-defined SVG filters

*This section is non-normative.*

Since drawing is performed using filter value "none" until an externally-defined filter has finished loading, authors might wish to determine whether such a filter has finished loading before proceeding with a drawing operation. One way to accomplish this is to load the externally-defined filter elsewhere within the same page in some element that sends a `load` event (for example, an [SVG use](#) element), and wait for the `load` event to be dispatched.

#### 4.12.5.1.21 Drawing model

When a shape or image is painted, user agents must follow these steps, in the order given (or act as if they do):

1. Render the shape or image onto an infinite [transparent black](#) bitmap, creating image *A*, as described in the previous sections. For shapes, the current fill, stroke, and line styles must be honored, and the stroke must itself also be subjected to the current transformation matrix.
2. When the [current filter](#) is set to a value other than "none" and all the externally-defined filters it references, if any, are in documents that are currently loaded, then use image *A* as the input to the [current filter](#), creating image *B*. If the [current filter](#) is a string parseable as a [<filter-value-list>](#), then draw using the [current filter](#) in the same manner as SVG.

Otherwise, let *B* be an alias for *A*.

3. [When shadows are drawn](#), render the shadow from image *B*, using the current shadow styles, creating image *C*.
4. [When shadows are drawn](#), multiply the alpha component of every pixel in *C* by [global alpha](#).
5. [When shadows are drawn](#), composite *C* within the [clipping region](#) over the current [output bitmap](#) using the [current compositing and blending operator](#).
6. Multiply the alpha component of every pixel in *B* by [global alpha](#).
7. Composite *B* within the [clipping region](#) over the current [output bitmap](#) using the [current compositing and blending operator](#).

When compositing onto the [output bitmap](#), pixels that would fall outside of the [output bitmap](#) must be discarded.



#### 4.12.5.1.22 *Best practices*

When a canvas is interactive, authors should include [focusable](#) elements in the element's fallback content corresponding to each [focusable](#) part of the canvas, as in the [example above](#).

When rendering focus rings, to ensure that focus rings have the appearance of native focus rings, authors should use the [drawFocusIfNeeded\(\)](#) method, passing it the element for which a ring is being drawn. This method only draws the focus ring if the element is [focused](#), so that it can simply be called whenever drawing the element, without checking whether the element is focused or not first.

In addition to drawing focus rings, authors should use the [scrollPathIntoView\(\)](#) method when an element in the canvas is focused, to make sure it is visible on the screen (if applicable).

Authors should avoid implementing text editing controls using the [canvas](#) element. Doing so has a large number of disadvantages:

- Mouse placement of the caret has to be reimplemented.
- Keyboard movement of the caret has to be reimplemented (possibly across lines, for multiline text input).
- Scrolling of the text control has to be implemented (horizontally for long lines, vertically for multiline input).
- Native features such as copy-and-paste have to be reimplemented.
- Native features such as spell-checking have to be reimplemented.
- Native features such as drag-and-drop have to be reimplemented.
- Native features such as page-wide text search have to be reimplemented.
- Native features specific to the user, for example custom text services, have to be reimplemented. This is close to impossible since each user might have different services installed, and there is an unbounded set of possible such services.
- Bidirectional text editing has to be reimplemented.
- For multiline text editing, line wrapping has to be implemented for all relevant languages.
- Text selection has to be reimplemented.
- Dragging of bidirectional text selections has to be reimplemented.
- Platform-native keyboard shortcuts have to be reimplemented.

- Platform-native input method editors (IMEs) have to be reimplemented.
- Undo and redo functionality has to be reimplemented.
- Accessibility features such as magnification following the caret or selection have to be reimplemented.

This is a huge amount of work, and authors are most strongly encouraged to avoid doing any of it by instead using the [input](#) element, the [textarea](#) element, or the [contenteditable](#) attribute.

#### 4.12.5.1.23 Examples

*This section is non-normative.*

Here is an example of a script that uses canvas to draw [pretty glowing lines](#).

```
<canvas width="800" height="450"></canvas>
<script>

  var context =
document.getElementsByTagName('canvas')[0].getContext('2d');

  var lastX = context.canvas.width * Math.random();
  var lastY = context.canvas.height * Math.random();
  var hue = 0;
  function line() {
    context.save();
    context.translate(context.canvas.width/2,
context.canvas.height/2);
    context.scale(0.9, 0.9);
    context.translate(-context.canvas.width/2, -
context.canvas.height/2);
    context.beginPath();
    context.lineWidth = 5 + Math.random() * 10;
    context.moveTo(lastX, lastY);
    lastX = context.canvas.width * Math.random();
    lastY = context.canvas.height * Math.random();
    context.bezierCurveTo(context.canvas.width * Math.random(),
                           context.canvas.height * Math.random(),
                           context.canvas.width * Math.random(),
                           context.canvas.height * Math.random(),
```

```

        lastX, lastY);

    hue = hue + 10 * Math.random();
    context.strokeStyle = 'hsl(' + hue + ', 50%, 50%)';
    context.shadowColor = 'white';
    context.shadowBlur = 10;
    context.stroke();
    context.restore();
}
setInterval(line, 50);

function blank() {
    context.fillStyle = 'rgba(0,0,0,0.1)';
    context.fillRect(0, 0, context.canvas.width,
context.canvas.height);
}
setInterval(blank, 40);

</script>

```

The 2D rendering context for [canvas](#) is often used for sprite-based games. The following example demonstrates this:

Here is the source for this example:

```

<!DOCTYPE HTML>
<html lang="en">
<meta charset="utf-8">
<title>Blue Robot Demo</title>
<style>
    html { overflow: hidden; min-height: 200px; min-width: 380px; }
    body { height: 200px; position: relative; margin: 8px; }
    .buttons { position: absolute; bottom: 0px; left: 0px; margin:
4px; }
</style>
<canvas width="380" height="200"></canvas>
<script>
    var Landscape = function (context, width, height) {
        this.offset = 0;
        this.width = width;
        this.advance = function (dx) {
            this.offset += dx;
        };
    };

```

```

    this.horizon = height * 0.7;

    // This creates the sky gradient (from a darker blue to white at
    the bottom)

    this.sky = context.createLinearGradient(0, 0, 0, this.horizon);
    this.sky.addColorStop(0.0, 'rgb(55,121,179)');
    this.sky.addColorStop(0.7, 'rgb(121,194,245)');
    this.sky.addColorStop(1.0, 'rgb(164,200,214)');

    // this creates the grass gradient (from a darker green to a
    lighter green)

    this.earth = context.createLinearGradient(0, this.horizon, 0,
    height);

    this.earth.addColorStop(0.0, 'rgb(81,140,20)');
    this.earth.addColorStop(1.0, 'rgb(123,177,57)');

    this.paintBackground = function (context, width, height) {
        // first, paint the sky and grass rectangles
        context.fillStyle = this.sky;
        context.fillRect(0, 0, width, this.horizon);
        context.fillStyle = this.earth;
        context.fillRect(0, this.horizon, width, height-this.horizon);

        // then, draw the cloudy banner

        // we make it cloudy by having the draw text off the top of
        the
        // canvas, and just having the blurred shadow shown on the
        canvas

        context.save();

        context.translate(width-((this.offset+(this.width*3.2)) %
        (this.width*4.0))+0, 0);

        context.shadowColor = 'white';

        context.shadowOffsetY = 30+this.horizon/3; // offset down on
        canvas

        context.shadowBlur = '5';
        context.fillStyle = 'white';
        context.textAlign = 'left';
        context.textBaseline = 'top';
        context.font = '20px sans-serif';

        context.fillText('WHATWG ROCKS', 10, -30); // text up above
        canvas

        context.restore();

        // then, draw the background tree

        context.save();

        context.translate(width-((this.offset+(this.width*0.2)) %
        (this.width*1.5))+30, 0);

        context.beginPath();

```

```

        context.fillStyle = 'rgb(143,89,2)';
        context.strokeStyle = 'rgb(10,10,10)';
        context.lineWidth = 2;
        context.rect(0, this.horizon+5, 10, -50); // trunk
        context.fill();
        context.stroke();
        context.beginPath();
        context.fillStyle = 'rgb(78,154,6)';
        context.arc(5, this.horizon-60, 30, 0, Math.PI*2); // leaves
        context.fill();
        context.stroke();
        context.restore();
    };

    this.paintForeground = function (context, width, height) {
        // draw the box that goes in front
        context.save();
        context.translate(width-((this.offset+(this.width*0.7)) %
(this.width*1.1))+0, 0);
        context.beginPath();
        context.rect(0, this.horizon - 5, 25, 25);
        context.fillStyle = 'rgb(220,154,94)';
        context.strokeStyle = 'rgb(10,10,10)';
        context.lineWidth = 2;
        context.fill();
        context.stroke();
        context.restore();
    };
};

</script>
<script>
var BlueRobot = function () {
    this.sprites = new Image();
    this.sprites.src = 'blue-robot.png'; // this sprite sheet has 8
cells
    this.targetMode = 'idle';
    this.walk = function () {
        this.targetMode = 'walk';
    };
    this.stop = function () {
        this.targetMode = 'idle';
    };
};

```

```

    this.frameIndex = {
        'idle': [0], // first cell is the idle frame
        'walk': [1,2,3,4,5,6], // the walking animation is cells 1-6
        'stop': [7], // last cell is the stopping animation
    };

    this.mode = 'idle';
    this.frame = 0; // index into frameIndex
    this.tick = function () {
        // this advances the frame and the robot
        // the return value is how many pixels the robot has moved
        this.frame += 1;

        if (this.frame >= this.frameIndex[this.mode].length) {
            // we've reached the end of this animation cycle
            this.frame = 0;

            if (this.mode != this.targetMode) {
                // switch to next cycle
                if (this.mode == 'walk') {
                    // we need to stop walking before we decide what to do
next
                    this.mode = 'stop';
                } else if (this.mode == 'stop') {
                    if (this.targetMode == 'walk')
                        this.mode = 'walk';
                    else
                        this.mode = 'idle';
                } else if (this.mode == 'idle') {
                    if (this.targetMode == 'walk')
                        this.mode = 'walk';
                }
            }

            if (this.mode == 'walk')
                return 8;
            return 0;
        },

        this.paint = function (context, x, y) {
            if (!this.sprites.complete) return;

            // draw the right frame out of the sprite sheet onto the
            canvas

            // we assume each frame is as high as the sprite sheet

```

```

    // the x,y coordinates give the position of the bottom center
    of the sprite
    context.drawImage(this.sprites,
        this.frameIndex[this.mode][this.frame] *
this.sprites.height, 0, this.sprites.height, this.sprites.height,
        x-this.sprites.height/2, y-
this.sprites.height, this.sprites.height, this.sprites.height);
    };
};
</script>
<script>
    var canvas = document.getElementsByTagName('canvas')[0];
    var context = canvas.getContext('2d');
    var landscape = new Landscape(context, canvas.width,
canvas.height);
    var blueRobot = new BlueRobot();
    // paint when the browser wants us to, using
    requestAnimationFrame()
    function paint() {
        context.clearRect(0, 0, canvas.width, canvas.height);
        landscape.paintBackground(context, canvas.width, canvas.height);
        blueRobot.paint(context, canvas.width/2, landscape.horizon*1.1);
        landscape.paintForeground(context, canvas.width, canvas.height);
        requestAnimationFrame(paint);
    }
    paint();
    // but tick every 100ms, so that we don't slow down when we don't
    paint
    setInterval(function () {
        var dx = blueRobot.tick();
        landscape.advance(dx);
    }, 100);
</script>
<p class="buttons">
    <input type="button" value="Walk" onclick="blueRobot.walk()">
    <input type="button" value="Stop" onclick="blueRobot.stop()">
</p>
<footer>
    <small> Blue Robot Player Sprite by <a
href="https://johncolburn.deviantart.com/">JohnColburn</a>.
    Licensed under the terms of the Creative Commons Attribution
    Share-Alike 3.0 Unported license.</small>
    <small> This work is itself licensed under a <a rel="license"
href="https://creativecommons.org/licenses/by-sa/3.0/">Creative

```

```
Commons Attribution-ShareAlike 3.0 Unported License</a>.</small>  
</footer>
```

#### 4.12.5.2 The [ImageBitmap](#) rendering context

##### 4.12.5.2.1 Introduction

[ImageBitmapRenderingContext](#) is a performance-oriented interface that provides a low overhead method for displaying the contents of [ImageBitmap](#) objects. It uses transfer semantics to reduce overall memory consumption. It also streamlines performance by avoiding intermediate compositing, unlike the [drawImage\(\)](#) method of [CanvasRenderingContext2D](#).

Using an [img](#) element as an intermediate for getting an image resource into a canvas, for example, would result in two copies of the decoded image existing in memory at the same time: the [img](#) element's copy, and the one in the canvas's backing store. This memory cost can be prohibitive when dealing with extremely large images. This can be avoided by using [ImageBitmapRenderingContext](#).

Using [ImageBitmapRenderingContext](#), here is how to transcode an image to the JPEG format in a memory- and CPU-efficient way:

```
createImageBitmap(inputImageBlob).then(image => {  
  const canvas = document.createElement('canvas');  
  const context = canvas.getContext('bitmaprenderer');  
  context.transferFromImageBitmap(image);  
  
  canvas.toBlob(outputJPEGBlob => {  
    // Do something with outputJPEGBlob.  
  }, 'image/jpeg');  
});
```

##### 4.12.5.2.2 The [ImageBitmapRenderingContext](#) interface



```
[Exposed=(Window,Worker)]
```

```
interface ImageBitmapRenderingContext {
```



```
readonly attribute (HTMLCanvasElement or OffscreenCanvas)
```

```
canvas;
```

```
undefined transferFromImageBitmap (ImageBitmap? bitmap);
```

```
};
```

```
dictionary ImageBitmapRenderingContextSettings {
```

```
    boolean alpha = true;
```

```
};
```

```
context = canvas.getContext('bitmaprenderer' [, { [ alpha: false ] } ])
```

Returns an [ImageBitmapRenderingContext](#) object that is permanently bound to a particular [canvas](#) element.

If the [alpha](#) setting is provided and set to false, then the canvas is forced to always be opaque.

```
context.canvas
```

Returns the [canvas](#) element that the context is bound to.

```
context.transferFromImageBitmap(imageBitmap)
```

✓MDN

Transfers the underlying [bitmap data](#) from *imageBitmap* to *context*, and the bitmap becomes the contents of the [canvas](#) element to which *context* is bound.

```
context.transferFromImageBitmap(null)
```

Replaces contents of the [canvas](#) element to which *context* is bound with a [transparent black](#) bitmap whose size corresponds to the [width](#) and [height](#) content attributes of the [canvas](#) element.

The **canvas** attribute must return the value it was initialized to when the object was created.

An [ImageBitmapRenderingContext](#) object has an **output bitmap**, which is a reference to [bitmap data](#).

An [ImageBitmapRenderingContext](#) object has a **bitmap mode**, which can be set to **valid** or **blank**. A value of [valid](#) indicates that the context's [output bitmap](#) refers to [bitmap data](#) that was acquired via [transferFromImageBitmap\(\)](#). A value [blank](#) indicates that the context's [output bitmap](#) is a default transparent bitmap.

An [ImageBitmapRenderingContext](#) object also has an **alpha** flag, which can be set to true or false. When an [ImageBitmapRenderingContext](#) object has its [alpha](#) flag set to false, the contents of the [canvas](#) element to which the context is bound are obtained by compositing the context's [output bitmap](#) onto an [opaque black](#) bitmap of the same size using the [source-over](#) compositing operator. If the [alpha](#) flag is set to true, then the [output bitmap](#) is used as the contents of the [canvas](#) element to which the context is bound. [\[COMPOSITE\]](#)

*The step of compositing over an [opaque black](#) bitmap ought to be elided whenever equivalent results can be obtained more efficiently by other means.*

---

When a user agent is required to **set an [ImageBitmapRenderingContext](#)'s output bitmap**, with a *context* argument that is an [ImageBitmapRenderingContext](#) object and an optional argument *bitmap* that refers to [bitmap data](#), it must run these steps:

1. If a *bitmap* argument was not provided, then:
  1. Set *context*'s [bitmap mode](#) to [blank](#).
  2. Let *canvas* be the [canvas](#) element to which *context* is bound.
  3. Set *context*'s [output bitmap](#) to be [transparent black](#) with an [intrinsic width](#) equal to [the numeric value](#) of *canvas*'s [width](#) attribute and an [intrinsic height](#) equal to [the numeric value](#) of *canvas*'s [height](#) attribute, those values being interpreted in [CSS pixels](#).
  4. Set the [output bitmap](#)'s [origin-clean](#) flag to true.
2. If a *bitmap* argument was provided, then:
  1. Set *context*'s [bitmap mode](#) to [valid](#).
  2. Set *context*'s [output bitmap](#) to refer to the same underlying bitmap data as *bitmap*, without making a copy.

*The [origin-clean](#) flag of bitmap is included in the bitmap data to be referenced by context's [output bitmap](#).*

---

The `ImageBitmapRenderingContext` **creation algorithm**, which is passed a *target* and *options*, consists of running these steps:

1. Let *settings* be the result of [converting](#) *options* to the dictionary type `ImageBitmapRenderingContextSettings`. (This can throw an exception.)
  2. Let *context* be a new `ImageBitmapRenderingContext` object.
  3. Initialize *context*'s `canvas` attribute to point to *target*.
  4. Set *context*'s [output bitmap](#) to the same bitmap as *target*'s bitmap (so that they are shared).
  5. Run the steps to [set an `ImageBitmapRenderingContext`'s output bitmap](#) with *context*.
  6. Initialize *context*'s `alpha` flag to true.
  7. Process each of the members of *settings* as follows:  

`alpha`  
If false, then set *context*'s `alpha` flag to false.
  8. Return *context*.
- 

The `transferFromImageBitmap(bitmap)` method, when invoked, must run these steps:

1. Let *bitmapContext* be the `ImageBitmapRenderingContext` object on which the `transferFromImageBitmap()` method was called.
2. If *bitmap* is null, then run the steps to [set an `ImageBitmapRenderingContext`'s output bitmap](#), with *bitmapContext* as the *context* argument and no *bitmap* argument, then return.
3. If the value of *bitmap*'s `[[Detached]]` internal slot is set to true, then throw an `"InvalidStateError" DOMException`.
4. Run the steps to [set an `ImageBitmapRenderingContext`'s output bitmap](#), with the *context* argument equal to *bitmapContext*, and the *bitmap* argument referring to *bitmap*'s underlying [bitmap data](#).
5. Set the value of *bitmap*'s `[[Detached]]` internal slot to true.
6. Unset *bitmap*'s [bitmap data](#).

### 4.12.5.3 The OffscreenCanvas interface

MDN

```
typedef (OffscreenCanvasRenderingContext2D or
```

```
ImageBitmapRenderingContext or WebGLRenderingContext or
```

```
WebGL2RenderingContext or GPUCanvasContext)
```

```
OffscreenRenderingContext;
```

```
dictionary ImageEncodeOptions {
```

```
    DOMString type = "image/png";
```

```
    unrestricted double quality;
```

```
};
```

```
enum OffscreenRenderingContextId { "2d", "bitmaprenderer",
```

```
"webgl", "webgl2", "webgpu" };
```

```
[Exposed=(Window,Worker), Transferable]
```

```
interface OffscreenCanvas : EventTarget {
```

```
    constructor([EnforceRange] unsigned long long width,
```

```
[EnforceRange] unsigned long long height);
```

```
    attribute [EnforceRange] unsigned long long width;
```

```
    attribute [EnforceRange] unsigned long long height;
```

```
    OffscreenRenderingContext?
```

```
    getContext(OffscreenRenderingContextId contextId, optional any
```

```
options = null);
```

```
ImageBitmap transferToImageBitmap();
```

```
Promise<Blob> convertToBlob(optional ImageEncodeOptions options  
= {});
```

```
attribute EventHandler oncontextlost;
```

```
attribute EventHandler oncontextrestored;
```

```
};
```

OffscreenCanvas is an EventTarget, so both OffscreenCanvasRenderingContext2D and WebGL can fire events at it. OffscreenCanvasRenderingContext2D can fire contextlost and contextrestored, and WebGL can fire webglcontextlost and webglcontextrestored. [\[WEBGL\]](#)

OffscreenCanvas objects are used to create rendering contexts, much like an HTMLCanvasElement, but with no connection to the DOM. This makes it possible to use canvas rendering contexts in [workers](#).



An OffscreenCanvas object may hold a weak reference to a **placeholder canvas element**, which is typically in the DOM, whose embedded content is provided by the OffscreenCanvas object. The bitmap of the OffscreenCanvas object is pushed to the [placeholder canvas element](#) by calling the `commit()` method of the OffscreenCanvas object's rendering context. All rendering context types that can be created by an OffscreenCanvas object must implement a `commit()` method. The exact behavior of the commit method (e.g. whether it copies or transfers bitmaps) may vary, as defined by the rendering contexts' respective specifications. Only the [2D context for offscreen canvases](#) is defined in this specification.

```
offscreenCanvas = new OffscreenCanvas(width, height)
```



Returns a new OffscreenCanvas object that is not linked to a [placeholder canvas element](#), and whose bitmap's size is determined by the `width` and `height` arguments.

```
context = offscreenCanvas.getContext(contextId [, options ])
```



Returns an object that exposes an API for drawing on the OffscreenCanvas object. `contextId` specifies the desired API: "2d",

`"bitmaprenderer", "webgl", "webgl2", or "webgpu".` *options* is handled by that API.

This specification defines the `"2d"` context below, which is similar but distinct from the `"2d"` context that is created from a `canvas` element. The WebGL specifications define the `"webgl"` and `"webgl2"` contexts. *WebGPU* defines the `"webgpu"` context. [WEBGL] [WEBGPU]

Returns null if the canvas has already been initialized with another context type (e.g., trying to get a `"2d"` context after getting a `"webgl"` context).

An `OffscreenCanvas` object has an internal **bitmap** that is initialized when the object is created. The width and height of the `bitmap` are equal to the values of the `width` and `height` attributes of the `OffscreenCanvas` object. Initially, all the bitmap's pixels are `transparent black`.

An `OffscreenCanvas` object can have a rendering context bound to it. Initially, it does not have a bound rendering context. To keep track of whether it has a rendering context or not, and what kind of rendering context it is, an `OffscreenCanvas` object also has a **context mode**, which is initially `none` but can be changed to either `2d`, `bitmaprenderer`, `webgl`, `webgl2`, `webgpu`, or `detached` by algorithms defined in this specification.

The constructor `OffscreenCanvas(width, height)`, when invoked, must create a new `OffscreenCanvas` object with its `bitmap` initialized to a rectangular array of `transparent black` pixels of the dimensions specified by `width` and `height`; and its `width` and `height` attributes initialized to `width` and `height` respectively.

---

`OffscreenCanvas` objects are `transferable`. Their `transfer steps`, given *value* and *dataHolder*, are as follows:

1. If *value*'s `context mode` is not equal to `none`, then throw an `"InvalidStateError" DOMException`.
2. Set *value*'s `context mode` to `detached`.
3. Let *width* and *height* be the dimensions of *value*'s `bitmap`.
4. Unset *value*'s `bitmap`.
5. Set *dataHolder*.[[Width]] to *width* and *dataHolder*.[[Height]] to *height*.
6. Set *dataHolder*.[[PlaceholderCanvas]] to be a weak reference to *value*'s `placeholder canvas element`, if *value* has one, or null if it does not.

Their [transfer-receiving steps](#), given *dataHolder* and *value*, are:

1. Initialize *value*'s [bitmap](#) to a rectangular array of [transparent black](#) pixels with width given by *dataHolder*.[[Width]] and height given by *dataHolder*.[[Height]].
2. If *dataHolder*.[[PlaceholderCanvas]] is not null, set *value*'s [placeholder canvas element](#) to *dataHolder*.[[PlaceholderCanvas]] (while maintaining the weak reference semantics).

---

The [getContext\(contextId, options\)](#) method of an [OffscreenCanvas](#) object, when invoked, must run these steps:

1. If *options* is not an [object](#), then set *options* to null.
2. Set *options* to the result of [converting options](#) to a JavaScript value.
3. Run the steps in the cell of the following table whose column header matches this [OffscreenCanvas](#) object's [context mode](#) and whose row header matches *contextId*:

	<a href="#">none</a>	<a href="#">2d</a>	<a href="#">bitmap renderer</a>	<a href="#">webgl or webgl2</a>	<a href="#">webgpu</a>	<a href="#">detached</a>
"2d"	Follow the <a href="#">offscreen 2D context creation algorithm</a> defined in the section below, passing it this <a href="#">OffscreenCanvas</a> object and <i>options</i> , to obtain an <a href="#">OffscreenCanvasRenderingContext2D</a> object; if this does not throw an exception, then set this <a href="#">OffscreenCanvas</a> object's <a href="#">context mode</a> to <a href="#">2d</a> , and return the new <a href="#">OffscreenCanvasRenderingContext2D</a> object.	Return the same object as was returned the last time the method was	Return null.	Return null.	Return null.	Throw an <a href="#">"InvalidStateError" DOMException</a> .

	<a href="#">none</a>	<a href="#">2d</a>	<a href="#">bitmap renderer</a>	<a href="#">webgl or we bgl2</a>	<a href="#">web gpu</a>	<a href="#">detached</a>
		s inv oke d wit h this sa me firs t arg um ent.				
" <a href="#">bitmap renderer</a> "	Follow the <a href="#">ImageBitmapRenderingContext</a> <a href="#">creation algorithm</a> defined in the section above, passing it this <a href="#">OffscreenCanvas</a> object and <i>options</i> , to obtain an <a href="#">ImageBitmapRenderingContext</a> object; if this does not throw an exception, then set this <a href="#">OffscreenCanvas</a> object's <a href="#">context</a> <a href="#">mode</a> to <a href="#">bitmaprenderer</a> , and return the new <a href="#">ImageBitmapRenderingContext</a> object.	Ret urn nul l.	Retur n the same object as was return ed the last time the metho d was invok ed with this same first argum ent.	Retur n null.	Ret urn nul l.	Throw an " <a href="#">InvalidStateError</a> " <a href="#">DOMException</a> .
" <a href="#">webgl1</a> " or " <a href="#">webgl2</a> "	Follow the instructions given in the WebGL specifications' <i>Context Creation</i> sections to obtain either a <a href="#">WebGLRenderingContext</a> , <a href="#">WebGL2RenderingContext</a> , or null; if the returned value is null, then return null; otherwise, set this <a href="#">OffscreenCanvas</a> object's <a href="#">context</a>	Ret urn nul l.	Retur n null.	Retur n the same value as was return ed the last time the	Ret urn nul l.	Throw an " <a href="#">InvalidStateError</a> " <a href="#">DOMException</a> .



	<u>none</u>	<u>2d</u>	<u>bitmap renderer</u>	<u>webgl or webgl2</u>	<u>webgpu</u>	<u>detached</u>
	<p><u>mode</u> to <u>webgl</u> or <u>webgl2</u>, and return the <u>WebGLRenderingContext</u> or <u>WebGL2RenderingContext</u> object. <a href="#">[WEBGL]</a></p>			<p>method was invoked with this same first argument.</p>		
"webgpu"	<p>Follow the instructions given in <i>WebGPU</i>'s <a href="#">Canvas Rendering</a> section to obtain a <u>GPUCanvasContext</u> or null; if the returned value is null, then return null; otherwise, set this <u>OffscreenCanvas</u> object's <u>context mode</u> to <u>webgpu</u> and return the <u>GPUCanvasContext</u> object. <a href="#">[WEBGPU]</a></p>	Return null.	Return null.	Return null.	Return the same value as was returned the last time the method was invoked with this same first argument.	Throw an <a href="#">"InvalidStateError"</a> <a href="#">DOMException</a> .

---

```
offscreenCanvas.width [ = value ]
```

MDN

```
offscreenCanvas.height [ = value ]
```

MDN

These attributes return the dimensions of the [OffscreenCanvas](#) object's [bitmap](#).

They can be set, to replace the [bitmap](#) with a new, [transparent black](#) bitmap of the specified dimensions (effectively resizing it).

If either the [width](#) or [height](#) attributes of an [OffscreenCanvas](#) object are set (to a new value or to the same value as before) and the [OffscreenCanvas](#) object's [context mode](#) is [2d](#), then [reset the rendering context to its default state](#) and resize the [OffscreenCanvas](#) object's [bitmap](#) to the new values of the [width](#) and [height](#) attributes.

The resizing behavior for "[webgl](#)" and "[webgl2](#)" contexts is defined in the WebGL specifications. [\[WEBGL\]](#)

The resizing behavior for "[webgpu](#)" context is defined in *WebGPU*. [\[WEBGPU\]](#)

*If an [OffscreenCanvas](#) object whose dimensions were changed has a [placeholder canvas element](#), then the [placeholder canvas element](#)'s [intrinsic size](#) will only be updated via the [commit\(\)](#) method of the [OffscreenCanvas](#) object's rendering context.*

```
promise = offscreenCanvas.convertToBlob([options])
```

△MDN

Returns a promise that will fulfill with a new [Blob](#) object representing a file containing the image in the [OffscreenCanvas](#) object.

The argument, if provided, is a dictionary that controls the encoding options of the image file to be created. The [type](#) field specifies the file format and has a default value of "[image/png](#)"; that type is also used if the requested type isn't supported. If the image format supports variable quality (such as "[image/jpeg](#)"), then the [quality](#) field is a number in the range 0.0 to 1.0 inclusive indicating the desired quality level for the resulting image.

```
canvas.transferToImageBitmap()
```

MDN

Returns a newly created [ImageBitmap](#) object with the image in the [OffscreenCanvas](#) object. The image in the [OffscreenCanvas](#) object is replaced with a new blank image.

The `convertToBlob(options)` method, when invoked, must run the following steps:

1. If the value of this `OffscreenCanvas` object's `[[Detached]]` internal slot is set to true, then return [a promise rejected with](#) an `"InvalidStateError" DOMException`.
2. If this `OffscreenCanvas` object's `context mode` is `2d` and the rendering context's `bitmap`'s `origin-clean` flag is set to false, then return [a promise rejected with](#) a `"SecurityError" DOMException`.
3. If this `OffscreenCanvas` object's `bitmap` has no pixels (i.e., either its horizontal dimension or its vertical dimension is zero) then return [a promise rejected with](#) an `"IndexSizeError" DOMException`.
4. Let *bitmap* be a copy of this `OffscreenCanvas` object's `bitmap`.
5. Let *result* be a new promise object.
6. Run these steps [in parallel](#):
  1. Let *file* be [a serialization of \*bitmap\* as a file](#), with *options*'s `type` and `quality` if present.
  2. [Queue an element task](#) on the [canvas blob serialization task source](#) given the `canvas` element to run these steps:
    1. If *file* is null, then reject *result* with an `"EncodingError" DOMException`.
    2. Otherwise, resolve *result* with a new `Blob` object, created in the [relevant realm](#) of this `OffscreenCanvas` object, representing *file*. [\[FILEAPI\]](#)
7. Return *result*.

The `transferToImageBitmap()` method, when invoked, must run the following steps:

1. If the value of this `OffscreenCanvas` object's `[[Detached]]` internal slot is set to true, then throw an `"InvalidStateError" DOMException`.
2. If this `OffscreenCanvas` object's `context mode` is set to `none`, then throw an `"InvalidStateError" DOMException`.
3. Let *image* be a newly created `ImageBitmap` object that references the same underlying bitmap data as this `OffscreenCanvas` object's `bitmap`.
4. Set this `OffscreenCanvas` object's `bitmap` to reference a newly created bitmap of the same dimensions and color space as the previous bitmap, and with its pixels initialized to [transparent black](#), or [opaque black](#) if the rendering context's `alpha` flag is set to false.

This means that if the rendering context of this [OffscreenCanvas](#) is a [WebGLRenderingContext](#), the value of [preserveDrawingBuffer](#) will have no effect. [\[WEBGL\]](#)

#### 5. Return *image*.

The following are the [event handlers](#) (and their corresponding [event handler event types](#)) that must be supported, as [event handler IDL attributes](#), by all objects implementing the [OffscreenCanvas](#) interface:

<a href="#">Event handler</a>	<a href="#">Event handler event type</a>
<a href="#">oncontextlost</a>	<a href="#">contextlost</a>
<a href="#">oncontextrestored</a>	<a href="#">contextrestored</a>

#### 4.12.5.3.1 The offscreen 2D rendering context

##### MDN

```
[Exposed=(Window,Worker)]
```

```
interface OffscreenCanvasRenderingContext2D {
```

```
    undefined commit();
```

```
    readonly attribute OffscreenCanvas canvas;
```

```
};
```

```
OffscreenCanvasRenderingContext2D includes CanvasState;
```

```
OffscreenCanvasRenderingContext2D includes CanvasTransform;
```

```
OffscreenCanvasRenderingContext2D includes CanvasCompositing;
```

```
OffscreenCanvasRenderingContext2D includes CanvasImageSmoothing;
```

```
OffscreenCanvasRenderingContext2D includes
```

```
CanvasFillStrokeStyles;
```

```
OffscreenCanvasRenderingContext2D includes CanvasShadowStyles;
```

[OffscreenCanvasRenderingContext2D](#) includes [CanvasFilters](#);

[OffscreenCanvasRenderingContext2D](#) includes [CanvasRect](#);

[OffscreenCanvasRenderingContext2D](#) includes [CanvasDrawPath](#);

[OffscreenCanvasRenderingContext2D](#) includes [CanvasText](#);

[OffscreenCanvasRenderingContext2D](#) includes [CanvasDrawImage](#);

[OffscreenCanvasRenderingContext2D](#) includes [CanvasImageData](#);

[OffscreenCanvasRenderingContext2D](#) includes

[CanvasPathDrawingStyles](#);

[OffscreenCanvasRenderingContext2D](#) includes

[CanvasTextDrawingStyles](#);

[OffscreenCanvasRenderingContext2D](#) includes [CanvasPath](#);

The [OffscreenCanvasRenderingContext2D](#) object is a rendering context for drawing to the [bitmap](#) of an [OffscreenCanvas](#) object. It is similar to the [CanvasRenderingContext2D](#) object, with the following differences:

- there is no support for [user interface](#) features;
- its [canvas](#) attribute refers to an [OffscreenCanvas](#) object rather than a [canvas](#) element;
- it has a [commit\(\)](#) method for pushing the rendered image to the context's [OffscreenCanvas](#) object's [placeholder canvas element](#).

An [OffscreenCanvasRenderingContext2D](#) object has a **bitmap** that is initialized when the object is created.

The [bitmap](#) has an **origin-clean** flag, which can be set to true or false. Initially, when one of these bitmaps is created, its [origin-clean](#) flag must be set to true.

An [OffscreenCanvasRenderingContext2D](#) object also has an **alpha** flag, which can be set to true or false. Initially, when the context is created, its alpha flag must be set to true. When an [OffscreenCanvasRenderingContext2D](#) object has its [alpha](#) flag set to false, then its alpha channel must be fixed to 1.0 (fully opaque) for all pixels, and attempts to change the alpha component of any pixel must be silently ignored.

An [OffscreenCanvasRenderingContext2D](#) object also has a **color space** setting of type [PredefinedColorSpace](#). The color space for the context's [bitmap](#) is set to the context's [color space](#).

An [OffscreenCanvasRenderingContext2D](#) object has an **associated [OffscreenCanvas](#) object**, which is the [OffscreenCanvas](#) object from which the [OffscreenCanvasRenderingContext2D](#) object was created.

```
offscreenCanvasRenderingContext2D.commit\(\)
```

Copies the rendering context's [bitmap](#) to the bitmap of the [placeholder canvas element](#) of the [associated OffscreenCanvas object](#). The copy operation is synchronous. Calling this method is not needed for the transfer, since it happens automatically during the [event loop](#) execution.

```
offscreenCanvas = offscreenCanvasRenderingContext2D.canvas
```

Returns the [associated OffscreenCanvas object](#).

The **offscreen 2D context creation algorithm**, which is passed a *target* (an [OffscreenCanvas](#) object) and optionally some arguments, consists of running the following steps:

1. If the algorithm was passed some arguments, let *arg* be the first such argument. Otherwise, let *arg* be undefined.
2. Let *settings* be the result of [converting](#) *arg* to the dictionary type [CanvasRenderingContext2DSettings](#). (This can throw an exception.).
3. Let *context* be a new [OffscreenCanvasRenderingContext2D](#) object.
4. Set *context*'s [associated OffscreenCanvas object](#) to *target*.
5. If *settings*["[alpha](#)"] is false, then set *context*'s [alpha](#) flag to false.
6. Set *context*'s [color space](#) to *settings*["[colorSpace](#)"].
7. Set *context*'s [bitmap](#) to a newly created bitmap with the dimensions specified by the [width](#) and [height](#) attributes of *target*, and set *target*'s bitmap to the same bitmap (so that they are shared).
8. If *context*'s [alpha](#) flag is set to true, initialize all the pixels of *context*'s [bitmap](#) to [transparent black](#). Otherwise, initialize the pixels to [opaque black](#).
9. Return *context*.

MDN

The `commit()` method, when invoked, must run the following steps:

1. If this [OffscreenCanvasRenderingContext2D](#)'s [associated OffscreenCanvas object](#) does not have a [placeholder canvas element](#), then return.
2. Let *image* be a copy of this [OffscreenCanvasRenderingContext2D](#)'s [bitmap](#), including the value of its [origin-clean](#) flag.
3. [Queue an element task](#) on the [DOM manipulation task source](#) given the [placeholder canvas element](#) to set the [placeholder canvas element](#)'s [output bitmap](#) to be a reference to *image*.

*If image has different dimensions than the bitmap previously referenced as the [placeholder canvas element](#)'s [output bitmap](#), then this task will result in a change in the [placeholder canvas element](#)'s [intrinsic size](#), which can affect document layout.*

*Implementations are encouraged to short-circuit the graphics update steps of the [window event loop](#) for the purposes of updating the contents of a [placeholder canvas element](#) to the display. This could mean, for example, that the [commit\(\)](#) method can copy the bitmap contents directly to a graphics buffer that is mapped to the physical display location of the [placeholder canvas element](#). This or similar short-circuiting approaches can significantly reduce display latency, especially in cases where the [commit\(\)](#) method is invoked from a [worker event loop](#) and the [window event loop](#) of the [placeholder canvas element](#) is busy. However, such shortcuts cannot have any script-observable side-effects. This means that the committed bitmap still needs to be sent to the [placeholder canvas element](#), in case the element is used as a [CanvasImageSource](#), as an [ImageBitmapSource](#), or in case [toDataURL\(\)](#) or [toBlob\(\)](#) are called on it.*

The **canvas** attribute, on getting, must return this [OffscreenCanvasRenderingContext2D](#)'s [associated OffscreenCanvas object](#).

#### 4.12.5.4 Color spaces and color space conversion

The [canvas](#) APIs provide mechanisms for specifying the color space of the canvas's backing store. The default backing store color space for all canvas APIs is '[srgb](#)'.

Color space conversion must be applied to the canvas's backing store when rendering the canvas to the output device. This color space conversion must be identical to the color space conversion that would be applied to an [img](#) element with a color profile that specifies the same [color space](#) as the canvas's backing store.

When drawing content to a 2D context, all inputs must be converted to the [context's color space](#) before drawing. Interpolation of gradient color stops must be performed

on color values after conversion to the [context's color space](#). Alpha blending must be performed on values after conversion to the [context's color space](#).

*There do not exist any inputs to a 2D context for which the color space is undefined. The color space for CSS colors is defined in CSS Color. The color space for images that specify no color profile information is assumed to be 'srgb', as specified in the [Color Spaces of Untagged Colors](#) section of CSS Color. [CSSCOLOR]*

#### 4.12.5.5 Serializing bitmaps to a file

When a user agent is to create **a serialization of the bitmap as a file**, given a *type* and an optional *quality*, it must create an image file in the format given by *type*. If an error occurs during the creation of the image file (e.g. an internal encoder error), then the result of the serialization is null. [PNG]

The image file's pixel data must be the bitmap's pixel data scaled to one image pixel per coordinate space unit, and if the file format used supports encoding resolution metadata, the resolution must be given as 96dpi (one image pixel per [CSS pixel](#)).

If *type* is supplied, then it must be interpreted as a [MIME type](#) giving the format to use. If the type has any parameters, then it must be treated as not supported.

For example, the value "[image/png](#)" would mean to generate a PNG image, the value "[image/jpeg](#)" would mean to generate a JPEG image, and the value "[image/svg+xml](#)" would mean to generate an SVG image (which would require that the user agent track how the bitmap was generated, an unlikely, though potentially awesome, feature).

User agents must support PNG ("[image/png](#)"). User agents may support other types. If the user agent does not support the requested type, then it must create the file using the PNG format. [PNG]

User agents must [convert the provided type to ASCII lowercase](#) before establishing if they support that type.

For image types that do not support an alpha channel, the serialized image must be the bitmap image composited onto an [opaque black](#) background using the [source-over](#) compositing operator.

For image types that support color profiles, the serialized image must include a color profile indicating the color space of the underlying bitmap. For image types that do not support color profiles, the serialized image must be converted to the ['srgb'](#) color space using ['relative-colorimetric'](#) rendering intent.

*Thus, in the 2D context, calling the [drawImage\(\)](#) method to render the output of the [toDataURL\(\)](#) or [toBlob\(\)](#) method to the canvas, given the appropriate*



*dimensions, has no visible effect beyond, at most, clipping colors of the canvas to a more narrow gamut.*

If *type* is an image format that supports variable quality (such as "[image/jpeg](#)"), *quality* is given, and *type* is not "[image/png](#)", then, if [Type](#)(*quality*) is Number, and *quality* is in the range 0.0 to 1.0 inclusive, the user agent must treat *quality* as the desired quality level. Otherwise, the user agent must use its default quality value, as if the *quality* argument had not been given.

*The use of type-testing here, instead of simply declaring quality as a Web IDL `double`, is a historical artifact.*

*Different implementations can have slightly different interpretations of "quality". When the quality is not specified, an implementation-specific default is used that represents a reasonable compromise between compression ratio, image quality, and encoding time.*

#### 4.12.5.6 Security with [canvas](#) elements

*This section is non-normative.*

**Information leakage** can occur if scripts from one [origin](#) can access information (e.g. read pixels) from images from another origin (one that isn't the [same](#)).

To mitigate this, bitmaps used with [canvas](#) elements and [ImageBitmap](#) objects are defined to have a flag indicating whether they are [origin-clean](#). All bitmaps start with their [origin-clean](#) set to true. The flag is set to false when cross-origin images are used.

The [toDataURL\(\)](#), [toBlob\(\)](#), and [getImageData\(\)](#) methods check the flag and will throw a "[SecurityError](#)" [DOMException](#) rather than leak cross-origin data.

The value of the [origin-clean](#) flag is propagated from a source [canvas](#) element's bitmap to a new [ImageBitmap](#) object by [createImageBitmap\(\)](#). Conversely, a destination [canvas](#) element's bitmap will have its [origin-clean](#) flags set to false by [drawImage](#) if the source image is an [ImageBitmap](#) object whose bitmap has its [origin-clean](#) flag set to false.

The flag can be reset in certain situations; for example, when changing the value of the [width](#) or the [height](#) content attribute of the [canvas](#) element to which a [CanvasRenderingContext2D](#) is bound, the bitmap is cleared and its [origin-clean](#) flag is reset.

When using an [ImageBitmapRenderingContext](#), the value of the [origin-clean](#) flag is propagated from [ImageBitmap](#) objects when they are transferred to the [canvas](#) via [transferFromImageBitmap\(\)](#).




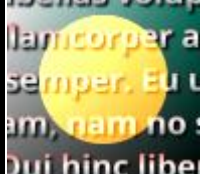
#### 4.12.5.7 Premultiplied alpha and the 2D rendering context

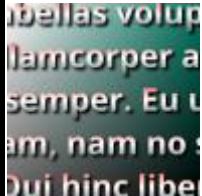
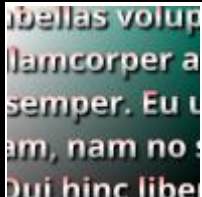
**Premultiplied alpha** refers to one way of representing transparency in an image, the other being non-premultiplied alpha.

Under non-premultiplied alpha, the red, green, and blue channels of a pixel represent that pixel's color, and its alpha channel represents that pixel's opacity.

Under premultiplied alpha, however, the red, green, and blue channels of a pixel represent the amounts of color that the pixel adds to the image, and its alpha channel represents the amount that the pixel obscures whatever is behind it.

For instance, assuming the color channels range from 0 (off) to 255 (full intensity), these example colors are represented in the following ways:

CSS color representation	Premultiplied representation	Non-premultiplied representation	Description of color	Image of color blended above other content
rgba(255, 127, 0, 1)	255, 127, 0, 255	255, 127, 0, 255	Completely-opaque orange	
rgba(255, 255, 0, 0.5)	127, 127, 0, 127	255, 255, 0, 127	Halfway-opaque yellow	
Unrepresentable	255, 127, 0, 127	Unrepresentable	Additive halfway-opaque orange	
Unrepresentable	255, 127, 0, 0	Unrepresentable	Additive fully-transparent orange	

CSS color representation	Premultiplied representation	Non-premultiplied representation	Description of color	Image of color blended above other content
rgba(255, 127, 0, 0)	0, 0, 0, 0	255, 127, 0, 0	Fully-transparent ("invisible") orange	
rgba(0, 127, 255, 0)	0, 0, 0, 0	255, 127, 0, 0	Fully-transparent ("invisible") turquoise	

**Converting a color value from a non-premultiplied representation to a premultiplied one** involves multiplying the color's red, green, and blue channels by its alpha channel (remapping the range of the alpha channel such that "fully transparent" is 0, and "fully opaque" is 1).

**Converting a color value from a premultiplied representation to a non-premultiplied one** involves the inverse: dividing the color's red, green, and blue channels by its alpha channel.

As certain colors can only be represented under premultiplied alpha (for instance, additive colors), and others can only be represented under non-premultiplied alpha (for instance, "invisible" colors which hold certain red, green, and blue values even with no opacity); and division and multiplication on 8-bit integers (which is how canvas's colors are currently stored) entails a loss of precision, converting between premultiplied and non-premultiplied alpha is a lossy operation on colors that are not fully opaque.

A [CanvasRenderingContext2D](#)'s [output bitmap](#) and an [OffscreenCanvasRenderingContext2D](#)'s [bitmap](#) must use premultiplied alpha to represent transparent colors.

*It is important for canvas bitmaps to represent colors using premultiplied alpha because it affects the range of representable colors. While additive colors cannot currently be drawn onto canvases directly because CSS colors are non-premultiplied and cannot represent them, it is still possible to, for instance, draw additive colors onto a WebGL canvas and then draw that WebGL canvas onto a 2D canvas via [drawImage\(\)](#).*

## 4.13 Custom elements

### 4.13.1 Introduction

*This section is non-normative.*

[Custom elements](#) provide a way for authors to build their own fully-featured DOM elements. Although authors could always use non-standard elements in their documents, with application-specific behavior added after the fact by scripting or similar, such elements have historically been non-conforming and not very functional. By [defining](#) a custom element, authors can inform the parser how to properly construct an element and how elements of that class should react to changes.

Custom elements are part of a larger effort to "rationalise the platform", by explaining existing platform features (like the elements of HTML) in terms of lower-level author-exposed extensibility points (like custom element definition). Although today there are many limitations on the capabilities of custom elements—both functionally and semantically—that prevent them from fully explaining the behaviors of HTML's existing elements, we hope to shrink this gap over time.

#### 4.13.1.1 Creating an autonomous custom element

*This section is non-normative.*

For the purposes of illustrating how to create an [autonomous custom element](#), let's define a custom element that encapsulates rendering a small icon for a country flag. Our goal is to be able to use it like so:

```
<flag-icon country="nl"></flag-icon>
```

To do this, we first declare a class for the custom element, extending [HTMLElement](#):

```
class FlagIcon extends HTMLElement {
  constructor() {
    super();
    this._countryCode = null;
  }

  static observedAttributes = ["country"];

  attributeChangedCallback(name, oldValue, newValue) {
    // name will always be "country" due to observedAttributes
    this._countryCode = newValue;
    this._updateRendering();
  }

  connectedCallback() {
```

```

    this._updateRendering();
}

get country() {
    return this._countryCode;
}

set country(v) {
    this.setAttribute("country", v);
}

_updateRendering() {
    // Left as an exercise for the reader. But, you'll probably
    want to
    // check this.ownerDocument.defaultView to see if we've been
    // inserted into a document with a browsing context, and avoid
    // doing any work if not.
}
}

```

We then need to use this class to define the element:

```
customElements.define("flag-icon", FlagIcon);
```

At this point, our above code will work! The parser, whenever it sees the `flag-icon` tag, will construct a new instance of our `FlagIcon` class, and tell our code about its new `country` attribute, which we then use to set the element's internal state and update its rendering (when appropriate).

You can also create `flag-icon` elements using the DOM API:

```

const flagIcon = document.createElement("flag-icon")
flagIcon.country = "jp"
document.body.appendChild(flagIcon)

```

Finally, we can also use the [custom element constructor](#) itself. That is, the above code is equivalent to:

```

const flagIcon = new FlagIcon()
flagIcon.country = "jp"
document.body.appendChild(flagIcon)

```

#### 4.13.1.2 Creating a form-associated custom element

*This section is non-normative.*

Adding a static `formAssociated` property, with a true value, makes an [autonomous custom element](#) a [form-associated custom element](#).

The [ElementInternals](#) interface helps you to implement functions and properties common to form control elements.

```
class MyCheckbox extends HTMLElement {
  static formAssociated = true;
  static observedAttributes = ['checked'];

  constructor() {
    super();
    this._internals = this.attachInternals();
    this.addEventListener('click', this._onClick.bind(this));
  }

  get form() { return this._internals.form; }
  get name() { return this.getAttribute('name'); }
  get type() { return this.localName; }

  get checked() { return this.hasAttribute('checked'); }
  set checked(flag) { this.toggleAttribute('checked', Boolean(flag)); }

  attributeChangedCallback(name, oldValue, newValue) {
    // name will always be "checked" due to observedAttributes
    this._internals.setFormValue(this.checked ? 'on' : null);
  }

  _onClick(event) {
    this.checked = !this.checked;
  }
}

customElements.define('my-checkbox', MyCheckbox);
```

You can use the custom element `my-checkbox` like a built-in form-associated element. For example, putting it in [form](#) or [label](#) associates the `my-checkbox` element with them, and submitting the [form](#) will send data provided by `my-checkbox` implementation.

```
<form action="..." method="...">
```

```
<label><my-checkbox name="agreed"></my-checkbox> I read the
agreement.</label>

<input type="submit">

</form>
```

#### 4.13.1.3 Creating a custom element with default accessible roles, states, and properties

*This section is non-normative.*

By using the appropriate properties of [ElementInternals](#), your custom element can have default accessibility semantics. The following code expands our form-associated checkbox from the previous section to properly set its default role and checkedness, as viewed by accessibility technology:

```
class MyCheckbox extends HTMLElement {
  static formAssociated = true;
  static observedAttributes = ['checked'];

  constructor() {
    super();
    this._internals = this.attachInternals();
    this.addEventListener('click', this._onClick.bind(this));

    this._internals.role = 'checkbox';
    this._internals.ariaChecked = 'false';
  }

  get form() { return this._internals.form; }
  get name() { return this.getAttribute('name'); }
  get type() { return this.localName; }

  get checked() { return this.hasAttribute('checked'); }
  set checked(flag) { this.toggleAttribute('checked',
    Boolean(flag)); }

  attributeChangedCallback(name, oldValue, newValue) {
    // name will always be "checked" due to observedAttributes
    this._internals.setFormValue(this.checked ? 'on' : null);
    this._internals.ariaChecked = this.checked;
  }
}
```

```

    _onClick(event) {
      this.checked = !this.checked;
    }
  }
}

customElements.define('my-checkbox', MyCheckbox);

```

Note that, like for built-in elements, these are only defaults, and can be overridden by the page author using the [role](#) and [aria-\\*](#) attributes:

```

<!-- This markup is non-conforming -->
<input type="checkbox" checked role="button" aria-checked="false">
<!-- This markup is probably not what the custom element author
intended -->
<my-checkbox role="button" checked aria-checked="false">

```

Custom element authors are encouraged to state what aspects of their accessibility semantics are strong native semantics, i.e., should not be overridden by users of the custom element. In our example, the author of the `my-checkbox` element would state that its [role](#) and [aria-checked](#) values are strong native semantics, thus discouraging code such as the above.

#### 4.13.1.4 Creating a customized built-in element

*This section is non-normative.*

[Customized built-in elements](#) are a distinct kind of [custom element](#), which are defined slightly differently and used very differently compared to [autonomous custom elements](#). They exist to allow reuse of behaviors from the existing elements of HTML, by extending those elements with new custom functionality. This is important since many of the existing behaviors of HTML elements can unfortunately not be duplicated by using purely [autonomous custom elements](#). Instead, [customized built-in elements](#) allow the installation of custom construction behavior, lifecycle hooks, and prototype chain onto existing elements, essentially "mixing in" these capabilities on top of the already-existing element.

[Customized built-in elements](#) require a distinct syntax from [autonomous custom elements](#) because user agents and other software key off an element's local name in order to identify the element's semantics and behavior. That is, the concept of [customized built-in elements](#) building on top of existing behavior depends crucially on the extended elements retaining their original local name.

In this example, we'll be creating a [customized built-in element](#) named `plastic-button`, which behaves like a normal button but gets fancy animation effects added



whenever you click on it. We start by defining a class, just like before, although this time we extend [HTMLButtonElement](#) instead of [HTMLElement](#):

```
class PlasticButton extends HTMLButtonElement {
  constructor() {
    super();

    this.addEventListener("click", () => {
      // Draw some fancy animation effects!
    });
  }
}
```

When defining our custom element, we have to also specify the `extends` option:

```
customElements.define("plastic-button", PlasticButton, { extends:
  "button" });
```

In general, the name of the element being extended cannot be determined simply by looking at what element interface it extends, as many elements share the same interface (such as [a](#) and [blockquote](#) both sharing [HTMLQuoteElement](#)).

To construct our [customized built-in element](#) from parsed HTML source text, we use the `is` attribute on a [button](#) element:

```
<button is="plastic-button">Click Me!</button>
```

Trying to use a [customized built-in element](#) as an [autonomous custom element](#) will *not* work; that is, `<plastic-button>Click me?</plastic-button>` will simply create an [HTMLElement](#) with no special behavior.

If you need to create a customized built-in element programmatically, you can use the following form of [createElement\(\)](#):

```
const plasticButton = document.createElement("button", { is:
  "plastic-button" });
plasticButton.textContent = "Click me!";
```

And as before, the constructor will also work:

```
const plasticButton2 = new PlasticButton();
console.log(plasticButton2.localName); // will output "button"
console.assert(plasticButton2 instanceof PlasticButton);
console.assert(plasticButton2 instanceof HTMLButtonElement);
```

Note that when creating a customized built-in element programmatically, the `is` attribute will not be present in the DOM, since it was not explicitly set. However, [it will be added to the output when serializing](#):

```
console.assert(!plasticButton.hasAttribute("is"));
console.log(plasticButton.outerHTML); // will output '<button
is="plastic-button"></button>'
```

Regardless of how it is created, all of the ways in which `button` is special apply to such "plastic buttons" as well: their focus behavior, ability to participate in [form submission](#), the `disabled` attribute, and so on.

[Customized built-in elements](#) are designed to allow extension of existing HTML elements that have useful user-agent supplied behavior or APIs. As such, they can only extend existing HTML elements defined in this specification, and cannot extend legacy elements such as `bgsound`, `blink`, `isindex`, `keygen`, `multicol`, `nextid`, or `spacer` that have been defined to use `HTMLUnknownElement` as their [element interface](#).

One reason for this requirement is future-compatibility: if a [customized built-in element](#) was defined that extended a currently-unknown element, for example `combobox`, this would prevent this specification from defining a `combobox` element in the future, as consumers of the derived [customized built-in element](#) would have come to depend on their base element having no interesting user-agent-supplied behavior.

#### 4.13.1.5 Drawbacks of autonomous custom elements

*This section is non-normative.*

As specified below, and alluded to above, simply defining and using an element called `taco-button` does not mean that such elements [represent](#) buttons. That is, tools such as web browsers, search engines, or accessibility technology will not automatically treat the resulting element as a button just based on its defined name.

To convey the desired button semantics to a variety of users, while still using an [autonomous custom element](#), a number of techniques would need to be employed:

- The addition of the `tabindex` attribute would make the `taco-button` [focusable](#). Note that if the `taco-button` were to become logically disabled, the `tabindex` attribute would need to be removed.
- The addition of an ARIA role and various ARIA states and properties helps convey semantics to accessibility technology. For example, setting the `role` to `"button"` will convey the semantics that this is a button, enabling users to

successfully interact with the control using usual button-like interactions in their accessibility technology. Setting the `aria-label` property is necessary to give the button an [accessible name](#), instead of having accessibility technology traverse its child text nodes and announce them. And setting the `aria-disabled` state to `"true"` when the button is logically disabled conveys to accessibility technology the button's disabled state.

- The addition of event handlers to handle commonly-expected button behaviors helps convey the semantics of the button to web browser users. In this case, the most relevant event handler would be one that proxies appropriate `keydown` events to become `click` events, so that you can activate the button both with keyboard and by clicking.
- In addition to any default visual styling provided for `taco-button` elements, the visual styling will also need to be updated to reflect changes in logical state, such as becoming disabled; that is, whatever style sheet has rules for `taco-button` will also need to have rules for `taco-button[disabled]`.

With these points in mind, a full-featured `taco-button` that took on the responsibility of conveying button semantics (including the ability to be disabled) might look something like this:

```
class TacoButton extends HTMLElement {
  static observedAttributes = ["disabled"];

  constructor() {
    super();
    this._internals = this.attachInternals();
    this._internals.role = "button";

    this.addEventListener("keydown", e => {
      if (e.code === "Enter" || e.code === "Space") {
        this.dispatchEvent(new PointerEvent("click", {
          bubbles: true,
          cancelable: true
        }));
      }
    });

    this.addEventListener("click", e => {
      if (this.disabled) {
        e.preventDefault();
        e.stopImmediatePropagation();
      }
    });
  }
}
```

```

    this._observer = new MutationObserver(() => {
        this._internals.ariaLabel = this.textContent;
    });
}

connectedCallback() {
    this.setAttribute("tabindex", "0");

    this._observer.observe(this, {
        childList: true,
        characterData: true,
        subtree: true
    });
}

disconnectedCallback() {
    this._observer.disconnect();
}

get disabled() {
    return this.hasAttribute("disabled");
}

set disabled(flag) {
    this.toggleAttribute("disabled", Boolean(flag));
}

attributeChangedCallback(name, oldValue, newValue) {
    // name will always be "disabled" due to observedAttributes
    if (this.disabled) {
        this.removeAttribute("tabindex");
        this._internals.ariaDisabled = "true";
    } else {
        this.setAttribute("tabindex", "0");
        this._internals.ariaDisabled = "false";
    }
}
}

```

Even with this rather-complicated element definition, the element is not a pleasure to use for consumers: it will be continually "sprouting" tabindex attributes of its own

volition, and its choice of `tabindex="0"` focusability behavior may not match the [button](#) behavior on the current platform. This is because as of now there is no way to specify default focus behavior for custom elements, forcing the use of the [tabindex](#) attribute to do so (even though it is usually reserved for allowing the consumer to override default behavior).

In contrast, a simple [customized built-in element](#), as shown in the previous section, would automatically inherit the semantics and behavior of the [button](#) element, with no need to implement these behaviors manually. In general, for any elements with nontrivial behavior and semantics that build on top of existing elements of HTML, [customized built-in elements](#) will be easier to develop, maintain, and consume.

#### 4.13.1.6 Upgrading elements after their creation

*This section is non-normative.*

Because [element definition](#) can occur at any time, a non-custom element could be [created](#), and then later become a [custom element](#) after an appropriate [definition](#) is registered. We call this process "upgrading" the element, from a normal element into a custom element.

[Upgrades](#) enable scenarios where it may be preferable for [custom element definitions](#) to be registered after relevant elements have been initially created, such as by the parser. They allow progressive enhancement of the content in the custom element. For example, in the following HTML document the element definition for `img-viewer` is loaded asynchronously:

```
<!DOCTYPE html>
<html lang="en">
<title>Image viewer example</title>

<img-viewer filter="Kelvin">
  
</img-viewer>

<script src="js/elements/img-viewer.js" async></script>
```

The definition for the `img-viewer` element here is loaded using a [script](#) element marked with the [async](#) attribute, placed after the `<img-viewer>` tag in the markup. While the script is loading, the `img-viewer` element will be treated as an undefined element, similar to a [span](#). Once the script loads, it will define the `img-viewer` element, and the existing `img-viewer` element on the page will be upgraded, applying the custom element's definition (which presumably includes applying an

image filter identified by the string "Kelvin", enhancing the image's visual appearance).

---

Note that [upgrades](#) only apply to elements in the document tree. (Formally, elements that are [connected](#).) An element that is not inserted into a document will stay un-upgraded. An example illustrates this point:

```
<!DOCTYPE html>
<html lang="en">
<title>Upgrade edge-cases example</title>

<example-element></example-element>

<script>
  "use strict";

  const inDocument = document.querySelector("example-element");
  const outOfDocument = document.createElement("example-element");

  // Before the element definition, both are HTMLElement:
  console.assert(inDocument instanceof HTMLElement);
  console.assert(outOfDocument instanceof HTMLElement);

  class ExampleElement extends HTMLElement {}
  customElements.define("example-element", ExampleElement);

  // After element definition, the in-document element was
  upgraded:
  console.assert(inDocument instanceof ExampleElement);
  console.assert(!(outOfDocument instanceof ExampleElement));

  document.body.appendChild(outOfDocument);

  // Now that we've moved the element into the document, it too was
  upgraded:
  console.assert(outOfDocument instanceof ExampleElement);
</script>
```

### 4.13.2 Requirements for custom element constructors and reactions

When authoring [custom element constructors](#), authors are bound by the following conformance requirements:

- A parameter-less call to `super()` must be the first statement in the constructor body, to establish the correct prototype chain and `this` value before any further code is run.
- A `return` statement must not appear anywhere inside the constructor body, unless it is a simple early-return (`return` or `return this`).
- The constructor must not use the `document.write()` or `document.open()` methods.
- The element's attributes and children must not be inspected, as in the non-[upgrade](#) case none will be present, and relying on upgrades makes the element less usable.
- The element must not gain any attributes or children, as this violates the expectations of consumers who use the `createElement` or `createElementNS` methods.
- In general, work should be deferred to `connectedCallback` as much as possible—especially work involving fetching resources or rendering. However, note that `connectedCallback` can be called more than once, so any initialization work that is truly one-time will need a guard to prevent it from running twice.
- In general, the constructor should be used to set up initial state and default values, and to set up event listeners and possibly a [shadow root](#).

Several of these requirements are checked during [element creation](#), either directly or indirectly, and failing to follow them will result in a custom element that cannot be instantiated by the parser or DOM APIs. This is true even if the work is done inside a constructor-initiated [microtask](#), as a [microtask checkpoint](#) can occur immediately after construction.

When authoring [custom element reactions](#), authors should avoid manipulating the node tree as this can lead to unexpected results.

An element's `connectedCallback` can be queued before the element is disconnected, but as the callback queue is still processed, it results in a `connectedCallback` for an element that is no longer connected:

```
class CParent extends HTMLElement {  
  connectedCallback() {  
    this.firstChild.remove();  
  }  
}
```

```

}
customElements.define("c-parent", CParent);

class CChild extends HTMLElement {
  connectedCallback() {
    console.log("CChild connectedCallback: isConnected =",
this.isConnected);
  }
}
customElements.define("c-child", CChild);

const parent = new CParent(),
      child = new CChild();
parent.append(child);
document.body.append(parent);

// Logs:
// CChild connectedCallback: isConnected = false

```

### 4.13.3 Core concepts

A **custom element** is an element that is [custom](#). Informally, this means that its constructor and prototype are defined by the author, instead of by the user agent. This author-supplied constructor function is called the **custom element constructor**.

Two distinct types of [custom elements](#) can be defined:

#### MDN

1. An **autonomous custom element**, which is defined with no `extends` option. These types of custom elements have a local name equal to their [defined name](#).
2. A **customized built-in element**, which is defined with an `extends` option. These types of custom elements have a local name equal to the value passed in their `extends` option, and their [defined name](#) is used as the value of the `is` attribute, which therefore must be a [valid custom element name](#).

After a [custom element](#) is [created](#), changing the value of the `is` attribute does not change the element's behavior, as it is saved on the element as its `is` [value](#).

[Autonomous custom elements](#) have the following element definition:



### **Categories:**

[Flow content](#).

[Phrasing content](#).

[Palpable content](#).

For [form-associated custom elements](#): [Listed](#), [labelable](#), [submittable](#), and [resettable form-associated element](#).

### **Contexts in which this element can be used:**

Where [phrasing content](#) is expected.

### **Content model:**

[Transparent](#).

### **Content attributes:**

[Global attributes](#), except the [is](#) attribute

[form](#), for [form-associated custom elements](#) — Associates the element with a [form](#) element

[disabled](#), for [form-associated custom elements](#) — Whether the form control is disabled

[readonly](#), for [form-associated custom elements](#) — Affects [willValidate](#), plus any behavior added by the custom element author

[name](#), for [form-associated custom elements](#) — Name of the element to use for [form submission](#) and in the [form.elements](#) API

Any other attribute that has no namespace (see prose).

### **Accessibility considerations:**

For [form-associated custom elements](#): [for authors](#); [for implementers](#).

Otherwise: [for authors](#); [for implementers](#).

### **DOM interface:**

Supplied by the element's author (inherits from [HTMLElement](#))

An [autonomous custom element](#) does not have any special meaning: it [represents](#) its children. A [customized built-in element](#) inherits the semantics of the element that it extends.

Any namespace-less attribute that is relevant to the element's functioning, as determined by the element's author, may be specified on an [autonomous custom element](#), so long as the attribute name is [XML-compatible](#) and contains no [ASCII upper alphas](#). The exception is the [is](#) attribute, which must not be specified on an [autonomous custom element](#) (and which will have no effect if it is).

[Customized built-in elements](#) follow the normal requirements for attributes, based on the elements they extend. To add custom attribute-based behavior, use [data-\\*](#) attributes.

---

An [autonomous custom element](#) is called a **form-associated custom element** if the element is associated with a [custom element definition](#) whose [form-associated](#) field is set to true.

The [name](#) attribute represents the [form-associated custom element](#)'s name.  
The [disabled](#) attribute is used to make the [form-associated custom element](#) non-interactive and to prevent its [submission value](#) from being submitted.  
The [form](#) attribute is used to explicitly associate the [form-associated custom element](#) with its [form owner](#).

The [readonly](#) attribute of [form-associated custom elements](#) specifies that the element is [barred from constraint validation](#). User agents don't provide any other behavior for the attribute, but custom element authors should, where possible, use its presence to make their control non-editable in some appropriate fashion, similar to the behavior for the [readonly](#) attribute on built-in form controls.

**Constraint validation:** If the [readonly](#) attribute is specified on a [form-associated custom element](#), the element is [barred from constraint validation](#).

The [reset algorithm](#) for [form-associated custom elements](#) is to [enqueue a custom element callback reaction](#) with the element, callback name "formResetCallback", and an empty argument list.

---

A **valid custom element name** is a sequence of characters *name* that meets all of the following requirements:

- *name* must match the [PotentialCustomElementName](#) production:

```
PotentialCustomElementName ::=
[a-z] (PCENChar) * '-' (PCENChar) *

PCENChar ::=
"-" | "." | [0-9] | "_" | [a-z] | #xB7 | [#xC0-#xD6] | [#xD8-#xF6] |
[#xF8-#x37D] | [#x37F-#x1FFF] | [#x200C-#x200D] | [#x203F-#x2040] |
[#x2070-#x218F] | [#x2C00-#x2FEF] | [#x3001-#xD7FF] | [#xF900-#xFDCE]
| [#xFDF0-#xFFFD] | [#x10000-#xEFFFF]
```

This uses the [EBNF notation](#) from the XML specification. [\[XML\]](#)

- *name* must not be any of the following:
  - annotation-xml
  - color-profile
  - font-face
  - font-face-src
  - font-face-uri
  - font-face-format

- o font-face-name
- o missing-glyph

The list of names above is the summary of all hyphen-containing element names from the [applicable specifications](#), namely SVG 2 and MathML. [\[SVG\]](#) [\[MATHML\]](#)

These requirements ensure a number of goals for [valid custom element names](#):

- They start with an [ASCII lower alpha](#), ensuring that the HTML parser will treat them as tags instead of as text.
- They do not contain any [ASCII upper alphas](#), ensuring that the user agent can always treat HTML elements ASCII-case-insensitively.
- They contain a hyphen, used for namespacing and to ensure forward compatibility (since no elements will be added to HTML, SVG, or MathML with hyphen-containing local names in the future).
- They can always be created with [createElement\(\)](#) and [createElementNS\(\)](#), which have restrictions that go beyond the parser's.

Apart from these restrictions, a large variety of names is allowed, to give maximum flexibility for use cases like `<math-α>` or `<emotion-😄>`.

A **custom element definition** describes a [custom element](#) and consists of:

#### A name

A [valid custom element name](#)

#### A local name

A local name

#### A constructor

A Web IDL [CustomElementConstructor](#) callback function type value wrapping the [custom element constructor](#)

#### A list of **observed attributes**

A `sequence<DOMString>`

#### A collection of **lifecycle callbacks**

A map, whose keys are the strings "connectedCallback", "disconnectedCallback", "adoptedCallback", "attributeChangedCallback", "formAssociatedCallback", "formDisabledCallback", "formResetCallback", and "formStateRestoreCallback". The corresponding values are either a Web IDL [Function](#) callback function type value, or null. By default the value of each entry is null.

#### A construction stack

A list, initially empty, that is manipulated by the [upgrade an element](#) algorithm and the [HTML element constructors](#). Each entry in the list will be either an element or an **already constructed** marker.

### A *form-associated* boolean

If this is true, user agent treats elements associated to this [custom element definition](#) as [form-associated custom elements](#).

### A *disable internals* boolean

Controls [attachInternals\(\)](#).

### A *disable shadow* boolean

Controls [attachShadow\(\)](#).

To look up a custom element definition, given a *document*, *namespace*, *localName*, and *is*, perform the following steps. They will return either a [custom element definition](#) or null:

1. If *namespace* is not the [HTML namespace](#), return null.
2. If *document*'s [browsing context](#) is null, return null.
3. Let *registry* be *document*'s [relevant global object](#)'s [CustomElementRegistry](#) object.
4. If there is a [custom element definition](#) in *registry* with [name](#) and [local name](#) both equal to *localName*, return that [custom element definition](#).
5. If there is a [custom element definition](#) in *registry* with [name](#) equal to *is* and [local name](#) equal to *localName*, return that [custom element definition](#).
6. Return null.

## 4.13.4 The [CustomElementRegistry](#) interface



Each [Window](#) object is associated with a unique instance of a [CustomElementRegistry](#) object, allocated when the [Window](#) object is created.

*Custom element registries are associated with [Window](#) objects, instead of [Document](#) objects, since each [custom element constructor](#) inherits from the [HTMLElement](#) interface, and there is exactly one [HTMLElement](#) interface per [Window](#) object.*



The `customElements` attribute of the [Window](#) interface must return the [CustomElementRegistry](#) object for that [Window](#) object.

```
[Exposed=Window]
```

```
interface CustomElementRegistry {
```

```
  [CEReactions] undefined define(DOMString name,
```

```
  CustomElementConstructor constructor, optional
```

```
  ElementDefinitionOptions options = {});
```

```
  (CustomElementConstructor or undefined) get(DOMString name);
```

```
  Promise<CustomElementConstructor> whenDefined(DOMString name);
```

```
  [CEReactions] undefined upgrade(Node root);
```

```
};
```

```
callback CustomElementConstructor = HTMLInputElement ();
```

```
dictionary ElementDefinitionOptions {
```

```
  DOMString extends;
```

```
};
```

Every [CustomElementRegistry](#) has a set of [custom element definitions](#), initially empty. In general, algorithms in this specification look up elements in the registry by any of [name](#), [local name](#), or [constructor](#).

Every [CustomElementRegistry](#) also has an **element definition is running** flag which is used to prevent reentrant invocations of [element definition](#). It is initially unset.

Every [CustomElementRegistry](#) also has a **when-defined promise map**, mapping [valid custom element names](#) to promises. It is used to implement the [whenDefined\(\)](#) method.

```
window.customElements.define(name, constructor)
```

✓MDN

Defines a new [custom element](#), mapping the given name to the given constructor as an [autonomous custom element](#).

```
window.customElements.define(name, constructor, {  
  extends: baseLocalName })
```

Defines a new [custom element](#), mapping the given name to the given constructor as a [customized built-in element](#) for the [element type](#) identified by the supplied *baseLocalName*. A ["NotSupportedError" DOMException](#) will be thrown upon trying to extend a [custom element](#) or an unknown element.

```
window.customElements.get(name)
```

✓MDN

Retrieves the [custom element constructor](#) defined for the given [name](#). Returns undefined if there is no [custom element definition](#) with the given [name](#).

```
window.customElements.whenDefined(name)
```

✓MDN

Returns a promise that will be fulfilled with the [custom element](#)'s constructor when a [custom element](#) becomes defined with the given name. (If such a [custom element](#) is already defined, the returned promise will be immediately fulfilled.) Returns a promise rejected with a ["SyntaxError" DOMException](#) if not given a [valid custom element name](#).

```
window.customElements.upgrade(root)
```

✓MDN

Tries to [upgrade](#) all [shadow-including inclusive descendant](#) elements of *root*, even if they are not [connected](#).

**Element definition** is a process of adding a [custom element definition](#) to the [CustomElementRegistry](#). This is accomplished by the [define\(\)](#) method. When invoked, the [define\(name, constructor, options\)](#) method must run these steps:

1. If [IsConstructor](#)(*constructor*) is false, then throw a [TypeError](#).
2. If *name* is not a [valid custom element name](#), then throw a ["SyntaxError" DOMException](#).
3. If this [CustomElementRegistry](#) contains an entry with [name](#) *name*, then throw a ["NotSupportedError" DOMException](#).
4. If this [CustomElementRegistry](#) contains an entry with [constructor](#) *constructor*, then throw a ["NotSupportedError" DOMException](#).
5. Let *localName* be *name*.
6. Let *extends* be the value of the `extends` member of *options*, or null if no such member exists.
7. If *extends* is not null, then:

1. If *extends* is a [valid custom element name](#), then throw a ["NotSupportedError" DOMException](#).
2. If the [element interface](#) for *extends* and the [HTML namespace](#) is [HTMLUnknownElement](#) (e.g., if *extends* does not indicate an element definition in this specification), then throw a ["NotSupportedError" DOMException](#).
3. Set *localName* to *extends*.
8. If this [CustomElementRegistry](#)'s [element definition is running](#) flag is set, then throw a ["NotSupportedError" DOMException](#).
9. Set this [CustomElementRegistry](#)'s [element definition is running](#) flag.
10. Let *formAssociated* be false.
11. Let *disableInternals* be false.
12. Let *disableShadow* be false.
13. Let *observedAttributes* be an empty `sequence<DOMString>`.
14. Run the following substeps while catching any exceptions:
  1. Let *prototype* be ? [Get](#)(*constructor*, "prototype").
  2. If [Type](#)(*prototype*) is not Object, then throw a [TypeError](#) exception.
  3. Let *lifecycleCallbacks* be a map with the keys "connectedCallback", "disconnectedCallback", "adoptedCallback", and "attributeChangedCallback", each of which belongs to an entry whose value is null.
  4. For each of the keys *callbackName* in *lifecycleCallbacks*, in the order listed in the previous step:
    1. Let *callbackValue* be ? [Get](#)(*prototype*, *callbackName*).
    2. If *callbackValue* is not undefined, then set the value of the entry in *lifecycleCallbacks* with key *callbackName* to the result of [converting](#) *callbackValue* to the Web IDL [Function](#) callback type. Rethrow any exceptions from the conversion.
  5. If the value of the entry in *lifecycleCallbacks* with key "attributeChangedCallback" is not null, then:
    1. Let *observedAttributesIterable* be ? [Get](#)(*constructor*, "observedAttributes").
    2. If *observedAttributesIterable* is not undefined, then set *observedAttributes* to the result

of [converting](#) *observedAttributesIterable* to a `sequence<DOMString>`. Rethrow any exceptions from the conversion.

6. Let *disabledFeatures* be an empty `sequence<DOMString>`.
7. Let *disabledFeaturesIterable* be ? [Get](#)(*constructor*, "disabledFeatures").
8. If *disabledFeaturesIterable* is not undefined, then set *disabledFeatures* to the result of [converting](#) *disabledFeaturesIterable* to a `sequence<DOMString>`. Rethrow any exceptions from the conversion.
9. Set *disableInternals* to true if *disabledFeatures* [contains](#) "internals".
10. Set *disableShadow* to true if *disabledFeatures* [contains](#) "shadow".
11. Let *formAssociatedValue* be ? [Get](#)( *constructor*, "formAssociated").
12. Set *formAssociated* to the result of [converting](#) *formAssociatedValue* to a `boolean`. Rethrow any exceptions from the conversion.
13. If *formAssociated* is true, for each of "formAssociatedCallback", "formResetCallback", "formDisabledCallback", and "formStateRestoreCallback" *callbackName*:
  1. Let *callbackValue* be ? [Get](#)(*prototype*, *callbackName*).
  2. If *callbackValue* is not undefined, then set the value of the entry in *lifecycleCallbacks* with key *callbackName* to the result of [converting](#) *callbackValue* to the Web IDL [Function](#) callback type. Rethrow any exceptions from the conversion.

Then, perform the following substep, regardless of whether the above steps threw an exception or not:

14. Unset this [CustomElementRegistry](#)'s [element definition is running](#) flag.

Finally, if the first set of substeps threw an exception, then rethrow that exception (thus terminating this algorithm). Otherwise, continue onward.

15. Let *definition* be a new [custom element definition](#) with [name](#) *name*, [local name](#) *localName*, [constructor](#) *constructor*, [observed attributes](#) *observedAttributes*, [lifecycle callbacks](#) *lifecycleCallbacks*, [form-associated](#) *formAssociated*, [disable internals](#) *disableInternals*, and [disable shadow](#) *disableShadow*.
16. Add *definition* to this [CustomElementRegistry](#).
17. Let *document* be this [CustomElementRegistry](#)'s [relevant global object](#)'s [associated Document](#).



18. Let *upgrade candidates* be all elements that are [shadow-including descendants](#) of *document*, whose namespace is the [HTML namespace](#) and whose local name is *localName*, in [shadow-including tree order](#). Additionally, if *extends* is non-null, only include elements whose [is value](#) is equal to *name*.
19. For each element *element* in *upgrade candidates*, [enqueue a custom element upgrade reaction](#) given *element* and *definition*.
20. If this [CustomElementRegistry](#)'s [when-defined promise map](#) contains an entry with key *name*:
  1. Let *promise* be the value of that entry.
  2. Resolve *promise* with *constructor*.
  3. Delete the entry with key *name* from this [CustomElementRegistry](#)'s [when-defined promise map](#).

When invoked, the [get\(name\)](#) method must run these steps:

1. If this [CustomElementRegistry](#) contains an entry with [name](#) *name*, then return that entry's [constructor](#).
2. Otherwise, return undefined.

When invoked, the [whenDefined\(name\)](#) method must run these steps:

1. If *name* is not a [valid custom element name](#), then return [a promise rejected with](#) a ["SyntaxError" DOMException](#).
2. If this [CustomElementRegistry](#) contains an entry with [name](#) *name*, then return [a promise resolved with](#) that entry's [constructor](#).
3. Let *map* be this [CustomElementRegistry](#)'s [when-defined promise map](#).
4. If *map* does not contain an entry with key *name*, create an entry in *map* with key *name* and whose value is a new promise.
5. Let *promise* be the value of the entry in *map* with key *name*.
6. Return *promise*.

The [whenDefined\(\)](#) method can be used to avoid performing an action until all appropriate [custom elements](#) are [defined](#). In this example, we combine it with the [:defined](#) pseudo-class to hide a dynamically-loaded article's contents until we're sure that all of the [autonomous custom elements](#) it uses are defined.

```
articleContainer.hidden = true;  
  
fetch(articleURL)
```

```

.then(response => response.text())
.then(text => {
  articleContainer.innerHTML = text;

  return Promise.all(
    [...articleContainer.querySelectorAll(":not(:defined)")]
      .map(el => customElements.whenDefined(el.localName))
  );
})
.then(() => {
  articleContainer.hidden = false;
});

```

When invoked, the `upgrade(root)` method must run these steps:

1. Let *candidates* be a [list](#) of all of *root*'s [shadow-including inclusive descendant](#) elements, in [shadow-including tree order](#).
2. [For each](#) *candidate* of *candidates*, [try to upgrade](#) *candidate*.

The `upgrade()` method allows upgrading of elements at will. Normally elements are automatically upgraded when they become [connected](#), but this method can be used if you need to upgrade before you're ready to connect the element.

```

const el = document.createElement("spider-man");

class SpiderMan extends HTMLElement {}
customElements.define("spider-man", SpiderMan);

console.assert(!(el instanceof SpiderMan)); // not yet upgraded

customElements.upgrade(el);
console.assert(el instanceof SpiderMan);    // upgraded!

```

#### 4.13.5 Upgrades

To **upgrade an element**, given as input a [custom element definition](#) *definition* and an element *element*, run the following steps:

1. If *element*'s [custom element state](#) is not "undefined" or "uncustomized", then return.

One scenario where this can occur due to reentrant invocation of this algorithm, as in the following example:

```
<!DOCTYPE html>
<x-foo id="a"></x-foo>
<x-foo id="b"></x-foo>

<script>
// Defining enqueues upgrade reactions for both "a" and "b"
customElements.define("x-foo", class extends HTMLElement {
  constructor() {
    super();

    const b = document.querySelector("#b");
    b.remove();

    // While this constructor is running for "a", "b" is
    still
    // undefined, and so inserting it into the document will
    enqueue a
    // second upgrade reaction for "b" in addition to the one
    enqueued
    // by defining x-foo.
    document.body.appendChild(b);
  }
})
</script>
```

This step will thus bail out the algorithm early when [upgrade an element](#) is invoked with "b" a second time.

2. Set *element's* [custom element definition](#) to *definition*.
3. Set *element's* [custom element state](#) to "failed".

*It will be set to "custom" [after the upgrade succeeds](#). For now, we set it to "failed" so that any reentrant invocations will hit [the above early-exit step](#).*

4. For each *attribute* in *element's* [attribute list](#), in order, [enqueue a custom element callback reaction](#) with *element*, callback name "attributeChangedCallback", and an argument list containing *attribute's* local name, null, *attribute's* value, and *attribute's* namespace.
5. If *element* is [connected](#), then [enqueue a custom element callback reaction](#) with *element*, callback name "connectedCallback", and an empty argument list.

6. Add *element* to the end of *definition*'s [construction stack](#).
7. Let *C* be *definition*'s [constructor](#).
8. Run the following substeps while catching any exceptions:
  1. If *definition*'s [disable shadow](#) is true and *element*'s [shadow root](#) is non-null, then throw a ["NotSupportedError"](#) [DOMException](#).

*This is needed as [attachShadow\(\)](#) does not use [look up a custom element definition](#) while [attachInternals\(\)](#) does.*

2. Set *element*'s [custom element state](#) to "precustomized".
3. Let *constructResult* be the result of [constructing](#) *C*, with no arguments.

*If *C* [non-conformantly](#) uses an API decorated with the [\[CEReactions\]](#) extended attribute, then the reactions enqueued at the beginning of this algorithm will execute during this step, before *C* finishes and control returns to this algorithm. Otherwise, they will execute after *C* and the rest of the upgrade process finishes.*

4. If [SameValue](#)(*constructResult*, *element*) is false, then throw a [TypeError](#).

*This can occur if *C* constructs another instance of the same custom element before calling `super()`, or if *C* uses JavaScript's `return-override` feature to return an arbitrary [HTMLElement](#) object from the constructor.*

9. Then, perform the following substep, regardless of whether the above steps threw an exception or not:

1. Remove the last entry from the end of *definition*'s [construction stack](#).

*Assuming *C* calls `super()` (as it will if it is [conformant](#)), and that the call succeeds, this will be the [already constructed marker](#) that replaced the element we pushed at the beginning of this algorithm. (The [HTML element constructor](#) carries out this replacement.)*

*If *C* does not call `super()` (i.e. it is not [conformant](#)), or if any step in the [HTML element constructor](#) throws, then this entry will still be *element*.*

10. Finally, if the above steps threw an exception, then:

1. Set *element*'s [custom element definition](#) to null.
2. Empty *element*'s [custom element reaction queue](#).
3. Rethrow the exception (thus terminating this algorithm).

If the above steps threw an exception, then [element's custom element state](#) will remain "failed" or "precustomized".

11. If *element* is a [form-associated custom element](#), then:

1. [Reset the form owner](#) of *element*. If *element* is associated with a [form](#) element, then [enqueue a custom element callback reaction](#) with *element*, callback name "formAssociatedCallback", and « the associated [form](#) ».
2. If *element* is [disabled](#), then [enqueue a custom element callback reaction](#) with *element*, callback name "formDisabledCallback" and « true ».

12. Set *element's* [custom element state](#) to "custom".

To **try to upgrade an element**, given as input an element *element*, run the following steps:

1. Let *definition* be the result of [looking up a custom element definition](#) given *element's* [node document](#), *element's* namespace, *element's* local name, and *element's* [is value](#).
2. If *definition* is not null, then [enqueue a custom element upgrade reaction](#) given *element* and *definition*.

#### 4.13.6 Custom element reactions

A [custom element](#) possesses the ability to respond to certain occurrences by running author code:

- When [upgraded](#), its [constructor](#) is run, with no arguments.
- When it [becomes connected](#), its `connectedCallback` is called, with no arguments.
- When it [becomes disconnected](#), its `disconnectedCallback` is called, with no arguments.
- When it is [adopted](#) into a new document, its `adoptedCallback` is called, given the old document and new document as arguments.
- When any of its attributes are [changed](#), [appended](#), [removed](#), or [replaced](#), its `attributeChangedCallback` is called, given the attribute's local name, old value, new value, and namespace as arguments. (An attribute's old or new value is considered to be null when the attribute is added or removed, respectively.)

- When the user agent [resets the form owner](#) of a [form-associated custom element](#) and doing so changes the form owner, its `formAssociatedCallback` is called, given the new form owner (or null if no owner) as an argument.
- When the form owner of a [form-associated custom element](#) is [reset](#), its `formResetCallback` is called.
- When the [disabled](#) state of a [form-associated custom element](#) is changed, its `formDisabledCallback` is called, given the new state as an argument.
- When user agent updates a [form-associated custom element](#)'s value on behalf of a user or [as part of navigation](#), its `formStateRestoreCallback` is called, given the new value and a string indicating a reason, "autocomplete" or "restore", as arguments.

We call these reactions collectively **custom element reactions**.

The way in which [custom element reactions](#) are invoked is done with special care, to avoid running author code during the middle of delicate operations. Effectively, they are delayed until "just before returning to user script". This means that for most purposes they appear to execute synchronously, but in the case of complicated composite operations (like [cloning](#), or [range](#) manipulation), they will instead be delayed until after all the relevant user agent processing steps have completed, and then run together as a batch.

Additionally, the precise ordering of these reactions is managed via a somewhat-complicated stack-of-queues system, described below. The intention behind this system is to guarantee that [custom element reactions](#) always are invoked in the same order as their triggering actions, at least within the local context of a single [custom element](#). (Because [custom element reaction](#) code can perform its own mutations, it is not possible to give a global ordering guarantee across multiple elements.)

---

Each [similar-origin window agent](#) has a **custom element reactions stack**, which is initially empty. A [similar-origin window agent](#)'s **current element queue** is the [element queue](#) at the top of its [custom element reactions stack](#). Each item in the stack is an **element queue**, which is initially empty as well. Each item in an [element queue](#) is an element. (The elements are not necessarily [custom](#) yet, since this queue is used for [upgrades](#) as well.)

Each [custom element reactions stack](#) has an associated **backup element queue**, which is an initially-empty [element queue](#). Elements are pushed onto the [backup element queue](#) during operations that affect the DOM without going through an API decorated with [\[CEReactions\]](#), or through the parser's [create an element for the token](#) algorithm. An example of this is a user-initiated editing operation which

modifies the descendants or attributes of an [editable](#) element. To prevent reentrancy when processing the [backup element queue](#), each [custom element reactions stack](#) also has a **processing the backup element queue** flag, initially unset.

All elements have an associated **custom element reaction queue**, initially empty. Each item in the [custom element reaction queue](#) is of one of two types:

- An **upgrade reaction**, which will [upgrade](#) the custom element and contains a [custom element definition](#); or
- A **callback reaction**, which will call a lifecycle callback, and contains a callback function as well as a list of arguments.

This is all summarized in the following schematic diagram:

To **enqueue an element on the appropriate element queue**, given an element *element*, run the following steps:

1. Let *reactionsStack* be *element*'s [relevant agent](#)'s [custom element reactions stack](#).
2. If *reactionsStack* is empty, then:
  1. Add *element* to *reactionsStack*'s [backup element queue](#).
  2. If *reactionsStack*'s [processing the backup element queue](#) flag is set, then return.
  3. Set *reactionsStack*'s [processing the backup element queue](#) flag.
  4. [Queue a microtask](#) to perform the following steps:
    1. [Invoke custom element reactions](#) in *reactionsStack*'s [backup element queue](#).
    2. Unset *reactionsStack*'s [processing the backup element queue](#) flag.
3. Otherwise, add *element* to *element*'s [relevant agent](#)'s [current element queue](#).

To **enqueue a custom element callback reaction**, given a [custom element](#) *element*, a callback name *callbackName*, and a list of arguments *args*, run the following steps:

1. Let *definition* be *element*'s [custom element definition](#).
2. Let *callback* be the value of the entry in *definition*'s [lifecycle callbacks](#) with key *callbackName*.

3. If *callback* is null, then return.
4. If *callbackName* is "attributeChangedCallback", then:
  1. Let *attributeName* be the first element of *args*.
  2. If *definition*'s [observed attributes](#) does not contain *attributeName*, then return.
5. Add a new [callback reaction](#) to *element*'s [custom element reaction queue](#), with callback function *callback* and arguments *args*.
6. [Enqueue an element on the appropriate element queue](#) given *element*.

To **enqueue a custom element upgrade reaction**, given an element *element* and [custom element definition](#) *definition*, run the following steps:

1. Add a new [upgrade reaction](#) to *element*'s [custom element reaction queue](#), with [custom element definition](#) *definition*.
2. [Enqueue an element on the appropriate element queue](#) given *element*.

To **invoke custom element reactions** in an [element queue](#) *queue*, run the following steps:

1. While *queue* is not [empty](#):
  1. Let *element* be the result of [dequeuing](#) from *queue*.
  2. Let *reactions* be *element*'s [custom element reaction queue](#).
  3. Repeat until *reactions* is empty:
    1. Remove the first element of *reactions*, and let *reaction* be that element. Switch on *reaction*'s type:

#### [upgrade reaction](#)

[Upgrade](#) *element* using *reaction*'s [custom element definition](#).

#### [callback reaction](#)

[Invoke](#) *reaction*'s callback function with *reaction*'s arguments, and with *element* as the [callback this value](#).

If this throws an exception, catch it, and [report the exception](#).

---



To ensure [custom element reactions](#) are triggered appropriately, we introduce the `[CEReactions]` IDL [extended attribute](#). It indicates that the relevant algorithm is to be supplemented with additional steps in order to appropriately track and invoke [custom element reactions](#).

The `[CEReactions]` extended attribute must take no arguments, and must not appear on anything other than an operation, attribute, setter, or deleter. Additionally, it must not appear on readonly attributes.

Operations, attributes, setters, or deleters annotated with the `[CEReactions]` extended attribute must run the following steps in place of the ones specified in their description:

1. [Push](#) a new [element queue](#) onto this object's [relevant agent](#)'s [custom element reactions stack](#).
2. Run the originally-specified steps for this construct, catching any exceptions. If the steps return a value, let *value* be the returned value. If they throw an exception, let *exception* be the thrown exception.
3. Let *queue* be the result of [popping](#) from this object's [relevant agent](#)'s [custom element reactions stack](#).
4. [Invoke custom element reactions](#) in *queue*.
5. If an exception *exception* was thrown by the original steps, rethrow *exception*.
6. If a value *value* was returned from the original steps, return *value*.

*The intent behind this extended attribute is somewhat subtle. One way of accomplishing its goals would be to say that every operation, attribute, setter, and deleter on the platform must have these steps inserted, and to allow implementers to optimize away unnecessary cases (where no DOM mutation is possible that could cause [custom element reactions](#) to occur).*

*However, in practice this imprecision could lead to non-interoperable implementations of [custom element reactions](#), as some implementations might forget to invoke these steps in some cases. Instead, we settled on the approach of explicitly annotating all relevant IDL constructs, as a way of ensuring interoperable behavior and helping implementations easily pinpoint all cases where these steps are necessary.*

Any nonstandard APIs introduced by the user agent that could modify the DOM in such a way as to cause [enqueueing a custom element callback reaction](#) or [enqueueing a custom element upgrade reaction](#), for example by modifying any attributes or child elements, must also be decorated with the `[CEReactions]` attribute.

*As of the time of this writing, the following nonstandard or not-yet-standardized APIs are known to fall into this category:*

- [HTMLInputElement](#)'s `webkitdirectory` and incremental IDL attributes
- [HTMLLinkElement](#)'s `scope` IDL attribute

#### 4.13.7 Element internals

Certain capabilities are meant to be available to a custom element author, but not to a custom element consumer. These are provided by the `element.attachInternals()` method, which returns an instance of `ElementInternals`. The properties and methods of `ElementInternals` allow control over internal features which the user agent provides to all elements.

##### `element.attachInternals()`

Returns an `ElementInternals` object targeting the custom element `element`. Throws an exception if `element` is not a custom element, if the "internals" feature was disabled as part of the element definition, or if it is called twice on the same element.

Each `HTMLElement` has an **attached internals** boolean, initially false.

MDN

The `attachInternals()` method steps are:

1. If `this`'s `is value` is not null, then throw a `"NotSupportedError" DOMException`.
2. Let `definition` be the result of [looking up a custom element definition](#) given `this`'s `node document`, its namespace, its local name, and null as the `is value`.
3. If `definition` is null, then throw an `"NotSupportedError" DOMException`.
4. If `definition`'s `disable internals` is true, then throw a `"NotSupportedError" DOMException`.
5. If `this`'s `attached internals` is true, then throw an `"NotSupportedError" DOMException`.
6. If `this`'s `custom element state` is not `"precustomized"` or `"custom"`, then throw a `"NotSupportedError" DOMException`.
7. Set `this`'s `attached internals` to true.
8. Return a new `ElementInternals` instance whose `target element` is `this`.

#### 4.13.7.1 The ElementInternals interface

MDN

The IDL for the ElementInternals interface is as follows, with the various operations and attributes defined in the following sections:

```
[Exposed=Window]
```

```
interface ElementInternals {
```

```
    // Shadow root access
```

```
    readonly attribute ShadowRoot? shadowRoot;
```

```
    // Form-associated custom elements
```

```
    undefined setFormValue((File or USVString or FormData)? value,
```

```
                           optional (File or USVString or
```

```
                           FormData)? state);
```

```
    readonly attribute HTMLFormElement? form;
```

```
    undefined setValidity(optional ValidityStateFlags flags = {},
```

```
                           optional DOMString message,
```

```
                           optional HTMLElement anchor);
```

```
    readonly attribute boolean willValidate;
```

```
    readonly attribute ValidityState validity;
```

```
    readonly attribute DOMString validationMessage;
```

```
    boolean checkValidity();
```

```
    boolean reportValidity();
```

```
readonly attribute NodeList labels;
```

```
};
```

```
// Accessibility semantics
```

```
ElementInternals includes ARIAMixin;
```

```
dictionary ValidityStateFlags {
```

```
    boolean valueMissing = false;
```

```
    boolean typeMismatch = false;
```

```
    boolean patternMismatch = false;
```

```
    boolean tooLong = false;
```

```
    boolean tooShort = false;
```

```
    boolean rangeUnderflow = false;
```

```
    boolean rangeOverflow = false;
```

```
    boolean stepMismatch = false;
```

```
    boolean badInput = false;
```

```
    boolean customError = false;
```

```
};
```

Each [ElementInternals](#) has a **target element**, which is a [custom element](#).

#### 4.13.7.2 Shadow root access

**[internals.shadowRoot](#)**

Returns the [ShadowRoot](#) for *internals*'s [target element](#), if the [target element](#) is a [shadow host](#), or null otherwise.

The `shadowRoot` getter steps are:

1. Let *target* be [this's target element](#).
2. If *target* is not a [shadow host](#), then return null.
3. Let *shadow* be *target's shadow root*.
4. If *shadow's available to element internals* is false, then return null.
5. Return *shadow*.

#### 4.13.7.3 Form-associated custom elements

**`internals.setFormValue(value)`**

Sets both the [state](#) and [submission value](#) of *internals's target element* to *value*.

If *value* is null, the element won't participate in form submission.

**`internals.setFormValue(value, state)`**

Sets the [submission value](#) of *internals's target element* to *value*, and its [state](#) to *state*.

If *value* is null, the element won't participate in form submission.

**`internals.form`**

Returns the [form owner](#) of *internals's target element*.

**`internals.setValidity(flags, message [, anchor ])`**

Marks *internals's target element* as suffering from the constraints indicated by the *flags* argument, and sets the element's validation message to *message*. If *anchor* is specified, the user agent might use it to indicate problems with the constraints of *internals's target element* when the [form owner](#) is validated interactively or [reportValidity\(\)](#) is called.

**`internals.setValidity({})`**

Marks *internals's target element* as [satisfying its constraints](#).

**`internals . willValidate`**

Returns true if *internals's target element* will be validated when the form is submitted; false otherwise.

**`internals.validity`**

Returns the [ValidityState](#) object for *internals's target element*.

**`internals . validationMessage`**

Returns the error message that would be shown to the user if *internals's target element* was to be checked for validity.

```
valid = internals . checkValidity()
```

Returns true if *internals's* [target element](#) has no validity problems; false otherwise. Fires an [invalid](#) event at the element in the latter case.

```
valid = internals . reportValidity()
```

Returns true if *internals's* [target element](#) has no validity problems; otherwise, returns false, fires an [invalid](#) event at the element, and (if the event isn't canceled) reports the problem to the user.

```
internals.labels
```

Returns a [NodeList](#) of all the [label](#) elements that *internals's* [target element](#) is associated with.

Each [form-associated custom element](#) has **submission value**. It is used to provide one or more [entries](#) on form submission. The initial value of [submission value](#) is null, and [submission value](#) can be null, a string, a [File](#), or a [list](#) of [entries](#).

Each [form-associated custom element](#) has **state**. It is information with which the user agent can restore a user's input for the element. The initial value of [state](#) is null, and [state](#) can be null, a string, a [File](#), or a [list](#) of [entries](#).

The [setFormValue\(\)](#) method is used by the custom element author to set the element's [submission value](#) and [state](#), thus communicating these to the user agent.

When the user agent believes it is a good idea to restore a [form-associated custom element's](#) [state](#), for example [after navigation](#) or restarting the user agent, they may [enqueue a custom element callback reaction](#) with that element, callback name "formStateRestoreCallback", an argument list containing the state to be restored, and "restore".

If the user agent has a form-filling assist feature, then when the feature is invoked, it may [enqueue a custom element callback reaction](#) with a [form-associated custom element](#), callback name "formStateRestoreCallback", an argument list containing the state value determined by history of state value and some heuristics, and "autocomplete".

In general, the [state](#) is information specified by a user, and the [submission value](#) is a value after canonicalization or sanitization, suitable for submission to the server. The following examples makes this concrete:

Suppose that we have a [form-associated custom element](#) which asks a user to specify a date. The user specifies "3/15/2019", but the control wishes to submit "2019-03-15" to the server. "3/15/2019" would be a [state](#) of the element, and "2019-03-15" would be a [submission value](#).

Suppose you develop a custom element emulating a the behavior of the existing [checkbox](#) [input](#) type. Its [submission value](#) would be the value of its `value` content attribute, or the string "on". Its [state](#) would be one of "checked", "unchecked", "checked/indeterminate", or "unchecked/indeterminate".

The `setFormValue(value, state)` method steps are:

1. Let *element* be [this](#)'s [target element](#).
  2. If *element* is not a [form-associated custom element](#), then throw a `"NotSupportedError"` [DOMException](#).
  3. Set [target element](#)'s [submission value](#) to *value* if *value* is not a [FormData](#) object, or to a [clone](#) of *value*'s [entry list](#) otherwise.
  4. If the *state* argument of the function is omitted, set *element*'s [state](#) to its [submission value](#).
  5. Otherwise, if *state* is a [FormData](#) object, set *element*'s [state](#) to a [clone](#) of *state*'s [entry list](#).
  6. Otherwise, set *element*'s [state](#) to *state*.
- 

Each [form-associated custom element](#) has validity flags named `valueMissing`, `typeMismatch`, `patternMismatch`, `tooLong`, `tooShort`, `rangeUnderflow`, `rangeOverflow`, `stepMismatch`, and `customError`. They are false initially.

Each [form-associated custom element](#) has a **validation message** string. It is the empty string initially.

Each [form-associated custom element](#) has a **validation anchor** element. It is null initially.

The `setValidity(flags, message, anchor)` method steps are:

1. Let *element* be [this](#)'s [target element](#).
2. If *element* is not a [form-associated custom element](#), then throw a `"NotSupportedError"` [DOMException](#).
3. If *flags* contains one or more true values and *message* is not given or is the empty string, then throw a [TypeError](#).
4. For each entry *flag* → *value* of *flags*, set *element*'s validity flag with the name *flag* to *value*.

5. Set *element*'s [validation message](#) to the empty string if *message* is not given or all of *element*'s validity flags are false, or to *message* otherwise.
6. If *element*'s `customError` validity flag is true, then set *element*'s [custom validity error message](#) to *element*'s [validation message](#). Otherwise, set *element*'s [custom validity error message](#) to the empty string.
7. Set *element*'s [validation anchor](#) to null if *anchor* is not given. Otherwise, if *anchor* is not a [shadow-including descendant](#) of *element*, then throw a `"NotFoundError"` [DOMException](#). Otherwise, set *element*'s [validation anchor](#) to *anchor*.

## MDN

The `validationMessage` getter steps are:

1. Let *element* be [this](#)'s [target element](#).
2. If *element* is not a [form-associated custom element](#), then throw a `"NotSupportedError"` [DOMException](#).
3. Return *element*'s [validation message](#).

The **entry construction algorithm** for a [form-associated custom element](#), given an element *element* and an [entry list](#) *entry list*, consists of the following steps:

1. If *element*'s [submission value](#) is a [list](#) of [entries](#), then [append](#) each item of *element*'s [submission value](#) to *entry list*, and return.

*In this case, user agent does not refer to the [name](#) content attribute value. An implementation of [form-associated custom element](#) is responsible to decide names of [entries](#). They can be the [name](#) content attribute value, they can be strings based on the [name](#) content attribute value, or they can be unrelated to the [name](#) content attribute.*

2. If the element does not have a [name](#) attribute specified, or its [name](#) attribute's value is the empty string, then return.
3. If the element's [submission value](#) is not null, [create an entry](#) with the [name](#) attribute value and the [submission value](#), and [append](#) it to *entry list*.

### 4.13.7.4 Accessibility semantics

`internals.role [ = value ]`

Sets or retrieves the default ARIA role for *internals*'s [target element](#), which will be used unless the page author overrides it using the [role](#) attribute.



```
internals.aria* [ = value ]
```

Sets or retrieves various default ARIA states or property values for *internals*'s [target element](#), which will be used unless the page author overrides them using the [aria-\\*](#) attributes.

Each [custom element](#) has a **native accessibility semantics map**, which is a [map](#), initially empty. See the [Requirements related to ARIA and to platform accessibility APIs](#) section for information on how this impacts platform accessibility APIs.

[ElementInternals](#) includes the [ARIAMixin](#) mixin. The accessors provided by this mixin are used to manipulate the [target element](#)'s [native accessibility semantics map](#), as follows:

The [ARIAMixin](#) [getter steps](#) for [ElementInternals](#), given *internals*, *idAttribute*, and *contentAttribute*, are:

1. Let *map* be *internals*'s [target element](#)'s [native accessibility semantics map](#).
2. If *map*[*contentAttribute*] [exists](#), then return it.
3. Return null.

The [ARIAMixin](#) [setter steps](#) for [ElementInternals](#), given *internals*, *idAttribute*, *contentAttribute*, and *value*, are:

1. Let *map* be *internals*'s [target element](#)'s [native accessibility semantics map](#).
2. If *value* is null, then [remove](#) *map*[*contentAttribute*].
3. Otherwise, [set](#) *map*[*contentAttribute*] to *value*.

## 4.14 Common idioms without dedicated elements

### 4.14.1 Breadcrumb navigation

This specification does not provide a machine-readable way of describing breadcrumb navigation menus. Authors are encouraged to just use a series of links in a paragraph. The [nav](#) element can be used to mark the section containing these paragraphs as being navigation blocks.

In the following example, the current page can be reached via two paths.

```
<nav>
  <p>
    <a href="/">Main</a> ▶
    <a href="/products/">Products</a> ▶
```

```

<a href="/products/dishwashers/">Dishwashers</a> ▶
<a>Second hand</a>
</p>
<p>
<a href="/">Main</a> ▶
<a href="/second-hand/">Second hand</a> ▶
<a>Dishwashers</a>
</p>
</nav>

```

### 4.14.2 Tag clouds

This specification does not define any markup specifically for marking up lists of keywords that apply to a group of pages (also known as *tag clouds*). In general, authors are encouraged to either mark up such lists using [u1](#) elements with explicit inline counts that are then hidden and turned into a presentational effect using a style sheet, or to use SVG.

Here, three tags are included in a short tag cloud:

```

<style>
.tag-cloud > li > span { display: none; }
.tag-cloud > li { display: inline; }
.tag-cloud-1 { font-size: 0.7em; }
.tag-cloud-2 { font-size: 0.9em; }
.tag-cloud-3 { font-size: 1.1em; }
.tag-cloud-4 { font-size: 1.3em; }
.tag-cloud-5 { font-size: 1.5em; }

@media speech {
  .tag-cloud > li > span { display:inline }
}
</style>
...
<ul class="tag-cloud">
  <li class="tag-cloud-4"><a title="28 instances"
href="/t/apple">apple</a> <span>(popular)</span>
  <li class="tag-cloud-2"><a title="6 instances"
href="/t/kiwi">kiwi</a> <span>(rare)</span>
  <li class="tag-cloud-5"><a title="41 instances"
href="/t/pear">pear</a> <span>(very popular)</span>

```

```
</ul>
```

The actual frequency of each tag is given using the `title` attribute. A CSS style sheet is provided to convert the markup into a cloud of differently-sized words, but for user agents that do not support CSS or are not visual, the markup contains annotations like "(popular)" or "(rare)" to categorize the various tags by frequency, thus enabling all users to benefit from the information.

The `ul` element is used (rather than `ol`) because the order is not particularly important: while the list is in fact ordered alphabetically, it would convey the same information if ordered by, say, the length of the tag.

The `tag rel`-keyword is *not* used on these `a` elements because they do not represent tags that apply to the page itself; they are just part of an index listing the tags themselves.

### 4.14.3 Conversations

This specification does not define a specific element for marking up conversations, meeting minutes, chat transcripts, dialogues in screenplays, instant message logs, and other situations where different players take turns in discourse.

Instead, authors are encouraged to mark up conversations using `p` elements and punctuation. Authors who need to mark the speaker for styling purposes are encouraged to use `span` or `b`. Paragraphs with their text wrapped in the `i` element can be used for marking up stage directions.

This example demonstrates this using an extract from Abbot and Costello's famous sketch, *Who's on first*:

```
<p> Costello: Look, you gotta first baseman?  
<p> Abbott: Certainly.  
<p> Costello: Who's playing first?  
<p> Abbott: That's right.  
<p> Costello becomes exasperated.  
<p> Costello: When you pay off the first baseman every month, who  
gets the money?  
<p> Abbott: Every dollar of it.
```

The following extract shows how an IM conversation log could be marked up, using the `data` element to provide Unix timestamps for each line. Note that the timestamps are provided in a format that the `time` element does not support, so the `data` element is used instead (namely, Unix `time_t` timestamps). Had the author wished to mark up the data using one of the date and time formats supported by the `time` element, that element could have been used instead of `data`. This could

be advantageous as it would allow data analysis tools to detect the timestamps unambiguously, without coordination with the page author.

```
<p> <data value="1319898155">14:22</data> <b>egof</b> I'm not that
nerdy, I've only seen 30% of the star trek episodes

<p> <data value="1319898192">14:23</data> <b>kaj</b> if you know
what percentage of the star trek episodes you have seen, you are
inarguably nerdy

<p> <data value="1319898200">14:23</data> <b>egof</b> it's
unarguably

<p> <data value="1319898228">14:23</data> <i>* kaj blinks</i>

<p> <data value="1319898260">14:24</data> <b>kaj</b> you are not
helping your case
```

HTML does not have a good way to mark up graphs, so descriptions of interactive conversations from games are more difficult to mark up. This example shows one possible convention using [dl](#) elements to list the possible responses at each point in the conversation. Another option to consider is describing the conversation in the form of a DOT file, and outputting the result as an SVG image to place in the document. [\[DOT\]](#)

```
<p> Next, you meet a fisher. You can say one of several greetings:
<dl>
  <dt> "Hello there!"
  <dd>
    <p> She responds with "Hello, how may I help you?"; you can
    respond with:
    <dl>
      <dt> "I would like to buy a fish."
      <dd> <p> She sells you a fish and the conversation finishes.
      <dt> "Can I borrow your boat?"
      <dd>
        <p> She is surprised and asks "What are you offering in
        return?".
        <dl>
          <dt> "Five gold." (if you have enough)
          <dt> "Ten gold." (if you have enough)
          <dt> "Fifteen gold." (if you have enough)
          <dd> <p> She lends you her boat. The conversation ends.
          <dt> "A fish." (if you have one)
          <dt> "A newspaper." (if you have one)
          <dt> "A pebble." (if you have one)
          <dd> <p> "No thanks", she replies. Your conversation options
          at this point are the same as they were after asking to borrow
          her boat, minus any options you've suggested before.
```

```

    </dl>
  </dd>
</dl>
</dd>
<dt> "Vote for me in the next election!"
<dd> <p> She turns away. The conversation finishes.
<dt> "Madam, are you aware that your fish are running away?"
<dd>
  <p> She looks at you skeptically and says "Fish cannot run,
miss".
<dl>
  <dt> "You got me!"
  <dd> <p> The fisher sighs and the conversation ends.
  <dt> "Only kidding."
  <dd> <p> "Good one!" she retorts. Your conversation options at
this
  point are the same as those following "Hello there!" above.
  <dt> "Oh, then what are they doing?"
  <dd> <p> She looks at her fish, giving you an opportunity to
steal
  her boat, which you do. The conversation ends.
</dl>
</dd>
</dl>

```

In some games, conversations are simpler: each character merely has a fixed set of lines that they say. In this example, a game FAQ/walkthrough lists some of the known possible responses for each character:

```

<section>
  <h1>Dialogue</h1>
  <p><small>Some characters repeat their lines in order each time
you interact
  with them, others randomly pick from amongst their lines. Those
who respond in
  order have numbered entries in the lists below.</small>
  <h2>The Shopkeeper</h2>
  <ul>
    <li>How may I help you?
    <li>Fresh apples!
    <li>A loaf of bread for madam?
  </ul>
  <h2>The pilot</h2>
  <p>Before the accident:

```

```

<ul>
  <li>I'm about to fly out, sorry!
  <li>Sorry, I'm just waiting for flight clearance and then I'll be off!
</ul>
<p>After the accident:
<ol>
  <li>I'm about to fly out, sorry!
  <li>Ok, I'm not leaving right now, my plane is being cleaned.
  <li>Ok, it's not being cleaned, it needs a minor repair first.
  <li>Ok, ok, stop bothering me! Truth is, I had a crash.
</ol>
<h2>Clan Leader</h2>
<p>During the first clan meeting:
<ul>
  <li>Hey, have you seen my daughter? I bet she's up to something nefarious again...
  <li>Nice weather we're having today, eh?
  <li>The name is Bailey, Jeff Bailey. How can I help you today?
  <li>A glass of water? Fresh from the well!
</ul>
<p>After the earthquake:
<ol>
  <li>Everyone is safe in the shelter, we just have to put out the fire!
  <li>I'll go and tell the fire brigade, you keep hosing it down!
</ol>
</section>

```

#### 4.14.4 Footnotes

HTML does not have a dedicated mechanism for marking up footnotes. Here are the suggested alternatives.

---

For short inline annotations, the [title](#) attribute could be used.

In this example, two parts of a dialogue are annotated with footnote-like content using the `title` attribute.

```
<p> <b>Customer</b>: Hello! I wish to register a complaint. Hello. Miss?
<p> <b>Shopkeeper</b>: <span title="Colloquial pronunciation of 'What do you'"
>Watcha</span> mean, miss?
<p> <b>Customer</b>: Uh, I'm sorry, I have a cold. I wish to make a complaint.
<p> <b>Shopkeeper</b>: Sorry, <span title="This is, of course, a lie.">we're
closing for lunch</span>.
```

*Unfortunately, relying on the `title` attribute is currently discouraged as many user agents do not expose the attribute in an accessible manner as required by this specification (e.g. requiring a pointing device such as a mouse to cause a tooltip to appear, which excludes keyboard-only users and touch-only users, such as anyone with a modern phone or tablet). If the `title` attribute is used, CSS can be used to draw the reader's attention to the elements with the attribute.*

For example, the following CSS places a dashed line below elements that have a `title` attribute.

```
[title] { border-bottom: thin dashed; }
```

---

For longer annotations, the `a` element should be used, pointing to an element later in the document. The convention is that the contents of the link be a number in square brackets.

In this example, a footnote in the dialogue links to a paragraph below the dialogue. The paragraph then reciprocally links back to the dialogue, allowing the user to return to the location of the footnote.

```
<p> Announcer: Number 16: The <i>hand</i>.
<p> Interviewer: Good evening. I have with me in the studio tonight
Mr Norman St John Polevaulter, who for the past few years has been
contradicting people. Mr Polevaulter, why <em>do</em> you
contradict people?
<p> Norman: I don't. <sup><a href="#fn1" id="r1">[1]</a></sup>
<p> Interviewer: You told me you did!
...
```

```
<section>
  <p id="fn1"><a href="#r1">[1]</a> This is, naturally, a lie,
  but paradoxically if it were true he could not say so without
  contradicting the interviewer and thus making it false.</p>
</section>
```

---

For side notes, longer annotations that apply to entire sections of the text rather than just specific words or sentences, the [aside](#) element should be used.

In this example, a sidebar is given after a dialogue, giving it some context.

```
<p> <span class="speaker">Customer</span>: I will not buy this
record, it is scratched.
<p> <span class="speaker">Shopkeeper</span>: I'm sorry?
<p> <span class="speaker">Customer</span>: I will not buy this
record, it is scratched.
<p> <span class="speaker">Shopkeeper</span>: No no no, this's'a
tobacconist's.
<aside>
  <p>In 1970, the British Empire lay in ruins, and foreign
  nationalists frequented the streets – many of them Hungarians
  (not the streets – the foreign nationals). Sadly, Alexander
  Yalt has been publishing incompetently-written phrase books.
</aside>
```

---

For figures or tables, footnotes can be included in the relevant [figcaption](#) or [caption](#) element, or in surrounding prose.

In this example, a table has cells with footnotes that are given in prose. A [figure](#) element is used to give a single legend to the combination of the table and its footnotes.

```
<figure>
  <figcaption>Table 1. Alternative activities for
  knights.</figcaption>
  <table>
```



```

<tr>
  <th> Activity
  <th> Location
  <th> Cost
<tr>
  <td> Dance
  <td> Wherever possible
  <td> £0<sup><a href="#fn1">1</a></sup>
<tr>
  <td> Routines, chorus scenes<sup><a href="#fn2">2</a></sup>
  <td> Undisclosed
  <td> Undisclosed
<tr>
  <td> Dining<sup><a href="#fn3">3</a></sup>
  <td> Camelot
  <td> Cost of ham, jam, and spam<sup><a href="#fn4">4</a></sup>
</table>
<p id="fn1">1. Assumed.</p>
<p id="fn2">2. Footwork impeccable.</p>
<p id="fn3">3. Quality described as "well".</p>
<p id="fn4">4. A lot.</p>
</figure>

```

## 4.15 Disabled elements

An element is said to be **actually disabled** if it is one of the following:

- a [button](#) element that is [disabled](#)
- an [input](#) element that is [disabled](#)
- a [select](#) element that is [disabled](#)
- a [textarea](#) element that is [disabled](#)
- an [optgroup](#) element that has a [disabled](#) attribute
- an [option](#) element that is [disabled](#)
- a [fieldset](#) element that is a [disabled fieldset](#)
- a [form-associated custom element](#) that is [disabled](#)

*This definition is used to determine what elements are [focusable](#) and which elements match the [:enabled](#) and [:disabled](#) [pseudo classes](#).*

## 4.16 Matching HTML elements using selectors and CSS

### 4.16.1 Case-sensitivity of the CSS ['attr\(\)'](#) function

*CSS Values and Units* leaves the case-sensitivity of attribute names for the purpose of the ['attr\(\)'](#) function to be defined by the host language. [\[CSSVALUES\]](#)

When comparing the attribute name part of a CSS ['attr\(\)'](#) function to the names of namespace-less attributes on [HTML elements](#) in [HTML documents](#), the name part of the CSS ['attr\(\)'](#) function must first be [converted to ASCII lowercase](#). The same function when compared to other attributes must be compared according to its original case. In both cases, to match the values must be [identical to](#) each other (and therefore the comparison is case sensitive).

*This is the same as comparing the name part of a CSS [attribute selector](#), specified in the next section.*

### 4.16.2 Case-sensitivity of selectors

*Selectors* leaves the case-sensitivity of element names, attribute names, and attribute values to be defined by the host language. [\[SELECTORS\]](#)

When comparing a CSS element [type selector](#) to the names of [HTML elements](#) in [HTML documents](#), the CSS element [type selector](#) must first be [converted to ASCII lowercase](#). The same selector when compared to other elements must be compared according to its original case. In both cases, to match the values must be [identical to](#) each other (and therefore the comparison is case sensitive).

When comparing the name part of a CSS [attribute selector](#) to the names of attributes on [HTML elements](#) in [HTML documents](#), the name part of the CSS [attribute selector](#) must first be [converted to ASCII lowercase](#). The same selector when compared to other attributes must be compared according to its original case. In both cases, the comparison is case-sensitive.

[Attribute selectors](#) on an [HTML element](#) in an [HTML document](#) must treat the *values* of attributes with the following names as [ASCII case-insensitive](#):

- `accept`
- `accept-charset`
- `align`
- `alink`

- axis
- bgcolor
- charset
- checked
- clear
- codetype
- color
- compact
- declare
- defer
- dir
- direction
- disabled
- enctype
- face
- frame
- hreflang
- http-equiv
- lang
- language
- link
- media
- method
- multiple
- nohref
- noresize
- noshade
- nowrap
- readonly
- rel
- rev
- rules
- scope
- scrolling
- selected
- shape
- target
- text
- type
- valign
- valuetype
- vlink

For example, the selector `[bgcolor="#ffffff"]` will match any HTML element with a `bgcolor` attribute with values including `#ffffff`, `#FFFFFF` and `#fffFFF`. This happens even if `bgcolor` has no effect for a given element (e.g., `div`).

The selector `[type=a s]` will match any HTML element with a `type` attribute whose value is `a`, but not whose value is `A`, due to the `s` flag.

All other attribute values and everything else must be treated as entirely identical to each other for the purposes of selector matching. This includes:

- [IDs](#) and [classes](#) in [no-quirks mode](#) and [limited-quirks mode](#)
- the names of elements not in the [HTML namespace](#)
- the names of [HTML elements](#) in [XML documents](#)
- the names of attributes of elements not in the [HTML namespace](#)
- the names of attributes of [HTML elements](#) in [XML documents](#)
- the names of attributes that themselves have namespaces

*Selectors defines that ID and class selectors (such as `#foo` and `.bar`), when matched against elements in documents that are in [quirks mode](#), will be matched in an [ASCII case-insensitive](#) manner. However, this does not apply for attribute selectors with "id" or "class" as the name part. The selector `[class="foobar"]` will treat its value as case-sensitive even in [quirks mode](#).*

### 4.16.3 Pseudo-classes

MDN

There are a number of dynamic selectors that can be used with HTML. This section defines when these selectors match HTML elements. [\[SELECTORS\]](#) [\[CSSUI\]](#)

**:defined**

✓MDN

The **:defined** [pseudo-class](#) must match any element that is [defined](#).

**:link**

✓MDN

**:visited**

✓MDN

All [a](#) elements that have an [href](#) attribute, and all [area](#) elements that have an [href](#) attribute, must match one of [:link](#) and [:visited](#).

Other specifications might apply more specific rules regarding how these elements are to match these [pseudo-classes](#), to mitigate some privacy concerns that apply with straightforward implementations of this requirement.

**:active**

✓MDN

The **:active** [pseudo-class](#) is defined to match an element while an element is **being activated** by the user.

To determine whether a particular element is [being activated](#) for the purposes of defining the **:active** [pseudo-class](#) only, an HTML user agent must use the first relevant entry in the following list.

If the element is a [button](#) element

If the element is an [input](#) element whose [type](#) attribute is in the [Submit Button](#), [Image Button](#), [Reset Button](#), or [Button](#) state

If the element is an [a](#) element that has an [href](#) attribute

If the element is an [area](#) element that has an [href](#) attribute

If the element is [focusable](#)

The element is [being activated](#) if it is [in a formal activation state](#).

For example, if the user is using a keyboard to push a [button](#) element by pressing the space bar, the element would match this [pseudo-class](#) in between the time that the element received the [keydown](#) event and the time the element received the [keyup](#) event.

If the element is [being actively pointed at](#)

The element is [being activated](#).

An element is said to be **in a formal activation state** between the time the user begins to indicate an intent to trigger the element's [activation behavior](#) and either the time the user stops indicating an intent to trigger the element's [activation behavior](#), or the time the element's [activation behavior](#) has finished running, whichever comes first.

An element is said to be **being actively pointed at** while the user indicates the element using a pointing device while that pointing device is in the "down" state (e.g. for a mouse, between the time the mouse button is pressed and the time it is depressed; for a finger in a multitouch environment, while the finger is touching the display surface).

*Per the definition in Selectors, [:active](#) also matches [flat tree](#) ancestors of elements that are [being activated](#). [\[SELECTORS\]](#)*

Additionally, any element that is the [labeled control](#) of a [label](#) element that is currently matching [:active](#), also matches [:active](#). (But, it does not count as being [being activated](#).)

[:hover](#)



The [:hover](#) [pseudo-class](#) is defined to match an element while the user **designates** an element with a pointing device. For the purposes of defining the [:hover](#) [pseudo-class](#) only, an HTML user agent must consider an element as being one that the user [designates](#) if it is an element that the user indicates using a pointing device.

*Per the definition in Selectors, [:hover](#) also matches [flat tree](#) ancestors of elements that are [designated](#). [\[SELECTORS\]](#)*

Additionally, any element that is the [labeled control](#) of a [label](#) element that is currently matching [:hover](#), also matches [:hover](#). (But, it does not count as being [designated](#).)

Consider in particular a fragment such as:

```
<p> <label for=c> <input id=a> </label> <span id=b> <input id=c> </span> </p>
```

If the user designates the element with ID "a" with their pointing device, then the `p` element (and all its ancestors not shown in the snippet above), the `label` element, the element with ID "a", and the element with ID "c" will match the `:hover` [pseudo-class](#). The element with ID "a" matches it by being [designated](#); the `label` and `p` elements match it because of the condition in *Selectors* about flat tree ancestors; and the element with ID "c" matches it through the additional condition above on [labeled controls](#) (i.e., its `label` element matches `:hover`). However, the element with ID "b" does *not* match `:hover`: its flat tree descendant is not designated, even though that flat tree descendant matches `:hover`.

#### `:focus`



For the purposes of the CSS `:focus` [pseudo-class](#), an **element has the focus** when:

- it is not itself a [navigable container](#); and
- at least one of the following is true:
  - it is one of the elements listed in the [current focus chain of the top-level traversable](#), or
  - its [shadow root](#) `shadowRoot` is not null and `shadowRoot` is the [root](#) of at least one element that [has the focus](#).

#### `:target`



For the purposes of the CSS `:target` [pseudo-class](#), the `Document`'s *target elements* are a list containing the `Document`'s [target element](#), if it is not null, or containing no elements, if it is. [\[SELECTORS\]](#)

#### `:open`

The `:open` [pseudo-class](#) is defined to match any [HTML element](#) whose `popover` attribute is not in the [no popover state](#) and whose [popover visibility state](#) is [showing](#).

#### `:closed`

The `:closed` [pseudo-class](#) is defined to match any [HTML element](#) whose `popover` attribute is not in the [no popover state](#) and whose [popover visibility state](#) is [hidden](#).

#### `:enabled`



The [`:enabled`](#) [pseudo-class](#) must match any [button](#), [input](#), [select](#), [textarea](#), [optgroup](#), [option](#), [fieldset](#) element, or [form-associated custom element](#) that is not [actually disabled](#).

[`:disabled`](#)



The [`:disabled`](#) [pseudo-class](#) must match any element that is [actually disabled](#).

[`:checked`](#)



The [`:checked`](#) [pseudo-class](#) must match any element falling into one of the following categories:

- [input](#) elements whose [type](#) attribute is in the [Checkbox](#) state and whose [checkedness](#) state is true
- [input](#) elements whose [type](#) attribute is in the [Radio Button](#) state and whose [checkedness](#) state is true
- [option](#) elements whose [selectedness](#) is true

[`:indeterminate`](#)



The [`:indeterminate`](#) [pseudo-class](#) must match any element falling into one of the following categories:

- [input](#) elements whose [type](#) attribute is in the [Checkbox](#) state and whose [indeterminate](#) IDL attribute is set to true
- [input](#) elements whose [type](#) attribute is in the [Radio Button](#) state and whose [radio button group](#) contains no [input](#) elements whose [checkedness](#) state is true.
- [progress](#) elements with no [value](#) content attribute

[`:default`](#)



The [`:default`](#) [pseudo-class](#) must match any element falling into one of the following categories:

- [Submit buttons](#) that are [default buttons](#) of their [form owner](#).
- [input](#) elements to which the [checked](#) attribute applies and that have a [checked](#) attribute

- [option](#) elements that have a [selected](#) attribute

#### **:placeholder-shown**

The [:placeholder-shown](#) [pseudo-class](#) must match any element falling into one of the following categories:

- [input](#) elements that have a [placeholder](#) attribute whose value is currently being presented to the user.
- [textarea](#) elements that have a [placeholder](#) attribute whose value is currently being presented to the user.

#### **:valid**



The [:valid](#) [pseudo-class](#) must match any element falling into one of the following categories:

- elements that are [candidates for constraint validation](#) and that [satisfy their constraints](#)
- [form](#) elements that are not the [form owner](#) of any elements that themselves are [candidates for constraint validation](#) but do not [satisfy their constraints](#)
- [fieldset](#) elements that have no descendant elements that themselves are [candidates for constraint validation](#) but do not [satisfy their constraints](#)

#### **:invalid**



The [:invalid](#) [pseudo-class](#) must match any element falling into one of the following categories:

- elements that are [candidates for constraint validation](#) but that do not [satisfy their constraints](#)
- [form](#) elements that are the [form owner](#) of one or more elements that themselves are [candidates for constraint validation](#) but do not [satisfy their constraints](#)
- [fieldset](#) elements that have of one or more descendant elements that themselves are [candidates for constraint validation](#) but do not [satisfy their constraints](#)

#### **:in-range**





The `:in-range` [pseudo-class](#) must match all elements that are [candidates for constraint validation](#), [have range limitations](#), and that are neither [suffering from an underflow](#) nor [suffering from an overflow](#).

#### `:out-of-range`



The `:out-of-range` [pseudo-class](#) must match all elements that are [candidates for constraint validation](#), [have range limitations](#), and that are either [suffering from an underflow](#) or [suffering from an overflow](#).

#### `:required`



The `:required` [pseudo-class](#) must match any element falling into one of the following categories:

- `input` elements that are [required](#)
- `select` elements that have a `required` attribute
- `textarea` elements that have a `required` attribute

#### `:optional`



The `:optional` [pseudo-class](#) must match any element falling into one of the following categories:

- `input` elements to which the `required` attribute applies that are not [required](#)
- `select` elements that do not have a `required` attribute
- `textarea` elements that do not have a `required` attribute

#### `:autofill`



#### `:-webkit-autofill`

The `:autofill` and `:-webkit-autofill` [pseudo-classes](#) must match `input` elements which have been autofilled by user agent. These pseudo-classes must stop matching if the user edits the autofilled field.

*One way such autofilling might happen is via the `autocomplete` attribute, but user agents could autofill even without that attribute being involved.*

#### `:read-only`



#### `:read-write`



The `:read-write` [pseudo-class](#) must match any element falling into one of the following categories, which for the purposes of Selectors are thus considered *user-alterable*: [\[SELECTORS\]](#)

- `input` elements to which the `readonly` attribute applies, and that are [mutable](#) (i.e. that do not have the `readonly` attribute specified and that are not [disabled](#))
- `textarea` elements that do not have a `readonly` attribute, and that are not [disabled](#)
- elements that are [editing hosts](#) or [editable](#) and are neither `input` elements nor `textarea` elements

The `:read-only` [pseudo-class](#) must match all other [HTML elements](#).

`:dir(ltr)`



The `:dir(ltr)` [pseudo-class](#) must match all elements whose [directionality](#) is `'ltr'`.

`:dir(rtl)`

The `:dir(rtl)` [pseudo-class](#) must match all elements whose [directionality](#) is `'rtl'`.

*This specification does not define when an element matches the `:lang()` dynamic [pseudo-class](#), as it is defined in sufficient detail in a language-agnostic fashion in [Selectors](#). [\[SELECTORS\]](#)*

## 5 Microdata

### 5.1 Introduction

#### 5.1.1 Overview

*This section is non-normative.*

Sometimes, it is desirable to annotate content with specific machine-readable labels, e.g. to allow generic scripts to provide services that are customized to the page, or to enable content from a variety of cooperating authors to be processed by a single script in a consistent manner.

For this purpose, authors can use the microdata features described in this section. Microdata allows nested groups of name-value pairs to be added to documents, in parallel with the existing content.

### 5.1.2 The basic syntax

*This section is non-normative.*

At a high level, microdata consists of a group of name-value pairs. The groups are called [items](#), and each name-value pair is a property. Items and properties are represented by regular elements.

To create an item, the [itemscope](#) attribute is used.

To add a property to an item, the [itemprop](#) attribute is used on one of the [item's](#) descendants.

Here there are two items, each of which has the property "name":

```
<div itemscope>
  <p>My name is <span itemprop="name">Elizabeth</span>.</p>
</div>

<div itemscope>
  <p>My name is <span itemprop="name">Daniel</span>.</p>
</div>
```

Markup without the microdata-related attributes does not have any effect on the microdata model.

These two examples are exactly equivalent, at a microdata level, as the previous two examples respectively:

```
<div itemscope>
  <p>My <em>name</em> is <span
itemprop="name">E<strong>liz</strong>abeth</span>.</p>
</div>

<section>
  <div itemscope>
    <aside>
      <p>My name is <span itemprop="name"><a
href="/?user=daniel">Daniel</a></span>.</p>
```

```
</aside>
</div>
</section>
```

Properties generally have values that are strings.

Here the item has three properties:

```
<div itemscope>
  <p>My name is <span itemprop="name">Neil</span>.</p>
  <p>My band is called <span itemprop="band">Four Parts
  Water</span>.</p>
  <p>I am <span itemprop="nationality">British</span>.</p>
</div>
```

When a string value is a [URL](#), it is expressed using the [a](#) element and its [href](#) attribute, the [img](#) element and its [src](#) attribute, or other elements that link to or embed external resources.

In this example, the item has one property, "image", whose value is a URL:

```
<div itemscope>
  
</div>
```

Lorsqu'une valeur de chaîne est dans un format lisible par machine impropre à la consommation humaine, elle est exprimée à l'aide de l' [value](#)attribut de l' [data](#) élément, avec la version lisible par l'homme indiquée dans le contenu de l'élément.

Ici, il y a un article avec une propriété dont la valeur est un identifiant de produit. L'ID n'est pas convivial, donc le nom du produit est utilisé dans le texte visible par l'homme au lieu de l'ID.

```
<h1 itemscope>
  <data itemprop="product-id" value="9678AOU879">The Instigator
  2000</data>
</h1>
```

Pour les données numériques, l' [meter](#)élément et son [value](#)attribut peuvent être utilisés à la place.

Ici, une note est donnée à l'aide d'un [meter](#)élément.

```
<div itemscope itemtype="http://schema.org/Product">
  <span itemprop="name">Panasonic White 60L Refrigerator</span>
  
```

```

<div itemprop="aggregateRating"
  itemscope itemType="http://schema.org/AggregateRating">
  <meter itemprop="ratingValue" min=0 value=3.5 max=5>Rated
  3.5/5</meter>
  (based on <span itemprop="reviewCount">11</span> customer
  reviews)
</div>
</div>

```

De même, pour les données liées à la date et à l'heure, l' [time](#) élément et son [datetime](#) attribut peuvent être utilisés à la place.

Dans cet exemple, l'élément a une propriété, "anniversaire", dont la valeur est une date :

```

<div itemscope>
  I was born on <time itemprop="birthday" datetime="2009-05-10">May
  10th 2009</time>.
</div>

```

Les propriétés peuvent également être elles-mêmes des groupes de paires nom-valeur, en mettant l' [itemscope](#) attribut sur l'élément qui déclare la propriété.

Les éléments qui ne font pas partie des autres sont appelés [éléments de microdonnées de niveau supérieur](#) .

Dans cet exemple, l'élément extérieur représente une personne et l'élément intérieur représente un groupe :

```

<div itemscope>
  <p>Name: <span itemprop="name">Amanda</span></p>
  <p>Band: <span itemprop="band" itemscope> <span
  itemprop="name">Jazz Band</span> (<span itemprop="size">12</span>
  players)</span></p>
</div>

```

L'élément extérieur a ici deux propriétés, "nom" et "bande". Le "nom" est "Amanda", et le "groupe" est un élément à part entière, avec deux propriétés, "nom" et "taille". Le "nom" du groupe est "Jazz Band", et la "taille" est "12".

L'élément externe dans cet exemple est un élément de microdonnées de niveau supérieur.

Les propriétés qui ne sont pas des descendants de l'élément avec l' [itemscope](#) attribut peuvent être associées à l' [élément](#) à l'aide de l' [itemref](#) attribut. Cet attribut prend une liste d'ID d'éléments à explorer en plus d'explorer les enfants de l'élément avec l' [itemscope](#) attribut.

Cet exemple est le même que le précédent, mais toutes les propriétés sont séparées de leurs [items](#) :

```
<div itemscope id="amanda" itemref="a b"></div>
<p id="a">Name: <span itemprop="name">Amanda</span></p>
<div id="b" itemprop="band" itemscope itemref="c"></div>
<div id="c">
  <p>Band: <span itemprop="name">Jazz Band</span></p>
  <p>Size: <span itemprop="size">12</span> players</p>
</div>
```

Cela donne le même résultat que l'exemple précédent. Le premier élément a deux propriétés, "name", définie sur "Amanda", et "band", définie sur un autre élément. Ce deuxième élément a deux autres propriétés, "name", défini sur "Jazz Band", et "size", défini sur "12".

Un [élément](#) peut avoir plusieurs propriétés avec le même nom et des valeurs différentes.

Cet exemple décrit une crème glacée, avec deux parfums :

```
<div itemscope>
  <p>Flavors in my favorite ice cream:</p>
  <ul>
    <li itemprop="flavor">Lemon sorbet</li>
    <li itemprop="flavor">Apricot sorbet</li>
  </ul>
</div>
```

Il en résulte ainsi un article à deux propriétés, toutes deux « saveur », ayant les valeurs « Sorbet citron » et « Sorbet abricot ».

Un élément introduisant une propriété peut également introduire plusieurs propriétés à la fois, pour éviter la duplication lorsque certaines propriétés ont la même valeur.

Ici, nous voyons un élément avec deux propriétés, "favorite-color" et "favorite-fruit", toutes deux définies sur la valeur "orange":

```
<div itemscope>
  <span itemprop="favorite-color favorite-fruit">orange</span>
</div>
```

Il est important de noter qu'il n'existe aucune relation entre les microdonnées et le contenu du document où les microdonnées sont annotées.

Il n'y a pas de différence sémantique, par exemple, entre les deux exemples suivants :

```
<figure>
  
  <figcaption><span itemscope><span itemprop="name">The
Castle</span></span> (1986)</figcaption>
</figure>
<span itemscope><meta itemprop="name" content="The Castle"></span>
<figure>
  
  <figcaption>The Castle (1986)</figcaption>
</figure>
```

Les deux ont une figure avec une légende, et les deux, sans aucun rapport avec la figure, ont un élément avec une paire nom-valeur avec le nom "nom" et la valeur "Le Château". La seule différence est que si l'utilisateur fait glisser la légende hors du document, dans le premier cas, l'élément sera inclus dans les données de glisser-déposer. Dans aucun des cas, l'image n'est associée de quelque manière que ce soit à l'article.

### 5.1.3 Éléments typés

*Cette section est non normative.*

Les exemples de la section précédente montrent comment des informations peuvent être balisées sur une page qui ne s'attend pas à ce que ses microdonnées soient réutilisées. Cependant, les microdonnées sont plus utiles lorsqu'elles sont utilisées dans des contextes où d'autres auteurs et lecteurs peuvent coopérer pour faire de nouvelles utilisations du balisage.

Pour cela, il est nécessaire de donner un type à chaque [élément](#), tel que "https://example.com/person", ou "https://example.org/cat", ou "https://band.example.filet/". Les types sont identifiés comme [des URL](#).

Le type d'un [élément](#) est donné comme la valeur d'un [itemtype](#)attribut sur le même élément que l' [itemscope](#)attribut.

Ici, le type d'élément est "https://example.org/animals#cat":

```
<section itemscope itemtype="https://example.org/animals#cat">
  <h1 itemprop="name">Hedral</h1>
  <p itemprop="desc">Hedral is a male american domestic
shorthair, with a fluffy black fur with white paws and belly.</p>
```

```

</section>
```

Dans cet exemple, l'élément "https://example.org/animals#cat" a trois propriétés, un "name" ("Hedral"), un "desc" ("Hedral is...") et un "img" ("hedral.jpeg").

Le type donne le contexte des propriétés, sélectionnant ainsi un vocabulaire : une propriété nommée « classe » donnée pour un élément de type « https://census.example/person » peut faire référence à la classe économique d'un individu, tandis qu'une La propriété nommée "classe" donnée pour un élément de type "https://example.com/school/teacher" peut faire référence à la classe à laquelle un enseignant a été affecté. Plusieurs types peuvent partager un vocabulaire. Par exemple, les types " https://example.org/people/teacher" et " https://example.org/people/engineer" pourraient être définis pour utiliser le même vocabulaire (bien que certaines propriétés ne soient peut-être pas particulièrement utiles dans les deux cas, par exemple peut-être que le https://example.org/people/engineer type " " pourrait ne pas être généralement utilisé avec le "classroom" propriété). Plusieurs types définis pour utiliser le même vocabulaire peuvent être donnés pour un seul élément en répertoriant les URL sous la forme d'une liste séparée par des espaces dans la valeur de l'attribut. Un élément ne peut pas recevoir deux types s'ils n'utilisent pas le même vocabulaire , cependant.

#### 5.1.4 Identifiants globaux pour les éléments

*Cette section est non normative.*

Parfois, un [élément](#) donne des informations sur un sujet qui a un identifiant global. Par exemple, les livres peuvent être identifiés par leur numéro ISBN.

Les vocabulaires (tels qu'identifiés par l' [itemtype](#) attribut) peuvent être conçus de sorte que [les éléments](#) soient associés à leur identifiant global de manière non ambiguë en exprimant les identifiants globaux sous forme [d'URL](#) données dans un [itemid](#) attribut.

La signification exacte des [URL](#) données dans [itemid](#) les attributs dépend du vocabulaire utilisé.

Ici, un article parle d'un livre en particulier :

```
<dl itemscope
  itemtype="https://vocab.example.net/book"
  itemid="urn:isbn:0-330-34032-8">
  <dt>Title
  <dd itemprop="title">The Reality Dysfunction
```



```

<dt>Author
<dd itemprop="author">Peter F. Hamilton
<dt>Publication date
<dd><time itemprop="pubdate" datetime="1996-01-26">26 January
1996</time>
</dl>

```

Le `https://vocab.example.net/book` vocabulaire " " dans cet exemple définirait que l' `itemid` attribut prend une `urn: URL` pointant vers l'ISBN du livre.

### 5.1.5 Sélection de noms lors de la définition de vocabulaires

*Cette section est non normative.*

Utiliser des microdonnées signifie utiliser un vocabulaire. À certaines fins, un vocabulaire ad hoc est suffisant. Pour d'autres, un vocabulaire devra être conçu. Dans la mesure du possible, les auteurs sont encouragés à réutiliser les vocabulaires existants, car cela facilite la réutilisation du contenu.

Lors de la conception de nouveaux vocabulaires, les identifiants peuvent être créés soit à l'aide d'[URL](#), soit, pour les propriétés, sous forme de mots simples (sans points ni deux-points). Pour les URL, les conflits avec d'autres vocabulaires peuvent être évités en n'utilisant que des identifiants qui correspondent aux pages sur lesquelles l'auteur a le contrôle.

Par exemple, si Jon et Adam écrivent tous les deux du contenu à `example.com`, à `https://example.com/~jon/...` et `https://example.com/~adam/...` respectivement, alors ils pourraient sélectionner des identifiants de la forme `"https://example.com/~jon/name"` et `"https://example.com/~adam /nom"` respectivement.

Les propriétés dont les noms sont simplement des mots ne peuvent être utilisées que dans le contexte des types auxquels elles sont destinées ; les propriétés nommées à l'aide d'URL peuvent être réutilisées dans des éléments de tout type. Si un élément n'a pas de type et ne fait pas partie d'un autre élément, alors si ses propriétés ont des noms qui ne sont que des mots simples, elles ne sont pas destinées à être globalement uniques et sont plutôt destinées à un usage limité. De manière générale, les auteurs sont encouragés à utiliser soit des propriétés avec des noms globalement uniques (URL), soit à s'assurer que leurs éléments sont typés.

Ici, un élément est un `"https://example.org/animals#cat"`, et la plupart des propriétés ont des noms qui sont des mots définis dans le contexte de ce type. Il existe également quelques propriétés supplémentaires dont les noms proviennent d'autres vocabulaires.

```

<section itemscope itemtype="https://example.org/animals#cat">

```

```

<h1 itemprop="name https://example.com/fn">Hedral</h1>
<p itemprop="desc">Hedral is a male American domestic
shorthair, with a fluffy <span
itemprop="https://example.com/color">black</span> fur with <span
itemprop="https://example.com/color">white</span> paws and
belly.</p>

</section>

```

Cet exemple a un élément avec le type "https://example.org/animals#cat" et les propriétés suivantes :

Propriété	Valeur
nom	hédrale
https://example.com/fn	hédrale
desc	Hedral est un mâle américain à poil court domestique, avec une fourrure noire duveteuse avec des pattes et un ventre blancs.
https://exemple.com/couleur	noir
https://exemple.com/couleur	blanc
image	.../hedral.jpeg

## 5.2 Encodage des microdonnées

### 5.2.1 Le modèle de microdonnées

Le modèle de microdonnées se compose de groupes de paires nom-valeur appelées [items](#) .

Chaque groupe est appelé un [élément](#) . Chaque [élément](#) peut avoir [des types d'éléments](#) , un [identificateur global](#) (si le vocabulaire spécifié par les [types d'éléments prend en charge les identificateurs globaux pour les éléments](#) ) et une liste de paires nom-valeur. Chaque nom de la paire nom-valeur est appelé [propriété](#) , et chaque [propriété](#) a une ou plusieurs [valeurs](#) . Chaque [valeur](#) est soit une chaîne, soit elle-même un groupe de paires nom-valeur (un [élément](#) ). Les noms ne sont pas ordonnés les uns par rapport aux autres, mais si un nom particulier a plusieurs valeurs, ils ont un ordre relatif.

### 5.2.2 Articles



Chaque [élément HTML](#) peut avoir un [itemscope](#)attribut spécifié. L' [itemscope](#)attribut est un [attribut booléen](#) .

Un élément avec l' [itemscope](#)attribut spécifié crée un nouvel **élément** , un groupe de paires nom-valeur.

---



Les éléments avec un [itemscope](#)attribut peuvent avoir un [itemtype](#)attribut spécifié, pour donner les [types d'éléments](#) de l' [élément](#) .

L' [itemtype](#)attribut, s'il est spécifié, doit avoir une valeur qui est un [ensemble non ordonné de jetons uniques séparés par des espaces](#) , dont aucun n'est [identique à](#) un autre jeton et dont chacun est une [chaîne d'URL valide](#) qui est une [URL absolue](#) , et qui sont tous définis pour utiliser le même vocabulaire. La valeur de l'attribut doit avoir au moins un jeton.

Les **types d'élément** d'un [élément](#) sont les jetons obtenus en [divisant la \[itemtype\]\(#\)valeur de l'attribut de l'élément sur l'espace blanc ASCII](#) . Si l' [itemtype](#)attribut est manquant ou si l'analyse de cette manière ne trouve aucun jeton, on dit que l' [élément](#) n'a pas [de type d'élément](#) .

Les [types d'éléments](#) doivent tous être des types définis dans [les spécifications applicables](#) et doivent tous être définis pour utiliser le même vocabulaire.

Sauf indication contraire dans cette spécification, les [URL](#) indiquées comme [types d'éléments](#) ne doivent pas être automatiquement déréférencées.

*Une spécification pourrait définir que son [type d'élément](#) peut être déréférencé pour fournir à l'utilisateur des informations d'aide, par exemple. En fait, les auteurs de vocabulaire sont encouragés à fournir des informations utiles à l' [URL](#) donnée .*

[Les types d'éléments](#) sont des identifiants opaques, et les agents utilisateurs ne doivent pas déréférencer [les types d'éléments](#) inconnus , ou autrement les déconstruire, afin de déterminer comment traiter [les éléments](#) qui les utilisent.

L' [itemtype](#)attribut ne doit pas être spécifié sur les éléments qui n'ont pas d' [itemscope](#)attribut spécifié.

---

Un [élément](#) est dit être un **élément typé** lorsqu'il a un [type d'élément](#) ou qu'il s'agit de la [valeur](#) d'une [propriété](#) d'un [élément typé](#) . Les **types pertinents** pour un [élément typé](#) sont les [types d'élément](#) de l'[élément](#) , s'il en a, ou bien sont les [types pertinents](#) de l' [élément](#) pour lequel il s'agit d'une [valeur](#) de [propriété](#) .

---



Les éléments avec un [itemscope](#)attribut et un [itemtype](#)attribut qui fait référence à un vocabulaire qui est défini pour **prendre en charge les identifiants globaux pour les éléments** peuvent également avoir un [itemid](#)attribut spécifié, pour donner un identifiant global pour l' [élément](#) , afin qu'il puisse être lié à d'autres [éléments](#) sur des pages ailleurs sur le la toile.

L' [itemid](#)attribut, s'il est spécifié, doit avoir une valeur correspondant à une [URL valide potentiellement entourée d'espaces](#) .

L' **identifiant global** d'un [élément](#) est la valeur de l'attribut de son élément [itemid](#), s'il en a un, [analysé](#) par rapport au [nœud document](#) de l'élément sur lequel l'attribut est spécifié. Si l' [itemid](#)attribut est manquant ou si sa résolution échoue, on dit qu'il n'a pas [d'identifiant global](#) .

L' [itemid](#)attribut ne doit pas être spécifié sur des éléments qui n'ont pas à la fois un [itemscope](#)attribut et un [itemtype](#)attribut spécifiés, et ne doit pas être spécifié sur des éléments avec un [itemscope](#)attribut dont [itemtype](#)l'attribut spécifie un vocabulaire qui ne prend pas [en charge les identifiants globaux pour les éléments](#) , comme défini par la spécification de ce vocabulaire .

La signification exacte d'un [identificateur global](#) est déterminée par la spécification du vocabulaire. Il appartient à ces spécifications de définir si plusieurs éléments avec le même identifiant global (que ce soit sur la même page ou sur des pages différentes) sont autorisés à exister, et quelles sont les règles de traitement pour ce vocabulaire en ce qui concerne la gestion du cas de plusieurs éléments avec le même identifiant.

---

Les éléments avec un `itemscope` attribut peuvent avoir un `itemref` attribut spécifié, pour donner une liste d'éléments supplémentaires à explorer pour trouver les paires nom-valeur de l' [élément](#) .

L' `itemref` attribut, s'il est spécifié, doit avoir une valeur qui est un [ensemble non ordonné de jetons uniques séparés par des espaces](#), dont aucun n'est [identique à](#) un autre jeton et composé d' [ID](#) d'éléments dans le même [arbre](#) .

L' `itemref` attribut ne doit pas être spécifié sur les éléments qui n'ont pas d' `itemscope` attribut spécifié.

*L' `itemref` attribut ne fait pas partie du modèle de données de microdonnées. Il s'agit simplement d'une construction syntaxique pour aider les auteurs à ajouter des annotations aux pages où les données à annoter ne suivent pas une structure arborescente pratique. Par exemple, il permet aux auteurs de baliser les données d'un tableau afin que chaque colonne définisse un [élément](#) distinct , tout en conservant les propriétés dans les cellules.*

Cet exemple montre un vocabulaire simple utilisé pour décrire les produits d'un fabricant de modélisme ferroviaire. Le vocabulaire n'a que cinq noms de propriétés :

#### **code produit**

Un entier qui nomme le produit dans le catalogue du fabricant.

#### **nom**

Une brève description du produit.

#### **escalader**

L'un des "HO", "1" ou "Z" (potentiellement avec un espace au début ou à la fin), indiquant l'échelle du produit.

#### **numérique**

S'il est présent, l'un des "Numérique", "Delta" ou "Systèmes" (potentiellement avec un espace au début ou à la fin) indiquant que le produit dispose d'un décodeur numérique du type donné.

#### **type de piste**

Pour les produits spécifiques à une piste, l'un des "K", "M", "C" (potentiellement avec un espace blanc au début ou à la fin) indiquant le type de piste pour lequel le produit est destiné.

Ce vocabulaire a quatre [types d'items](#) définis :

#### **`https://md.example.com/loco`**

Matériel roulant avec moteur

#### **`https://md.example.com/passagers`**

Matériel roulant voyageurs

<https://md.example.com/track>

Morceaux de piste

<https://md.example.com/lighting>

Equipement avec éclairage

Chaque [article](#) qui utilise ce vocabulaire peut se voir attribuer un ou plusieurs de ces types, selon le type de produit.

Ainsi, une locomotive peut être balisée comme suit :

```
<dl itemscope itemtype="https://md.example.com/loco
    https://md.example.com/lighting">
  <dt>Name:
  <dd itemprop="name">Tank Locomotive (DB 80)
  <dt>Product code:
  <dd itemprop="product-code">33041
  <dt>Scale:
  <dd itemprop="scale">HO
  <dt>Digital:
  <dd itemprop="digital">Delta
</dl>
```

Un kit de mise à niveau de lanterne d'aiguillage peut être marqué comme :

```
<dl itemscope itemtype="https://md.example.com/track
    https://md.example.com/lighting">
  <dt>Name:
  <dd itemprop="name">Turnout Lantern Kit
  <dt>Product code:
  <dd itemprop="product-code">74470
  <dt>Purpose:
  <dd>For retrofitting 2 <span itemprop="track-type">C</span> Track
    turnouts. <meta itemprop="scale" content="HO">
</dl>
```

Une voiture de tourisme sans éclairage peut être balisée comme suit :

```
<dl itemscope itemtype="https://md.example.com/passengers">
  <dt>Name:
  <dd itemprop="name">Express Train Passenger Car (DB Am 203)
  <dt>Product code:
  <dd itemprop="product-code">8710
```

```
<dt>Scale:  
<dd itemprop="scale">Z  
</dl>
```

Un grand soin est nécessaire lors de la création de nouveaux vocabulaires. Souvent, une approche hiérarchique des types peut être adoptée, ce qui se traduit par un vocabulaire où chaque élément n'a jamais qu'un seul type, ce qui est généralement beaucoup plus simple à gérer.

### 5.2.3 Noms : l' `itemprop` attribut



Chaque [élément HTML](#) peut avoir un `itemprop` attribut spécifié, si cela [ajoute une ou plusieurs propriétés](#) à un ou plusieurs [éléments](#) (comme défini ci-dessous).

L' `itemprop` attribut, s'il est spécifié, doit avoir une valeur qui est un [ensemble non ordonné de jetons uniques séparés par des espaces](#), dont aucun n'est [identique](#) à un autre jeton, représentant les noms des paires nom-valeur qu'il ajoute. La valeur de l'attribut doit avoir au moins un jeton.

Chaque jeton doit être soit :

- Si l'élément est un [élément typé](#) : un **nom de propriété défini** autorisé dans cette situation selon la spécification qui définit les [types pertinents](#) pour l'élément, ou
- Une [chaîne d'URL valide](#) qui est une [URL absolue](#) définie comme un nom de propriété d'élément autorisé dans cette situation par une spécification de vocabulaire, ou
- Une [chaîne d'URL valide](#) qui est une [URL absolue](#) , utilisée comme nom de propriété d'élément propriétaire (c'est-à-dire un nom utilisé par l'auteur à des fins privées, non défini dans une spécification publique), ou
- Si l'élément n'est pas un [élément typé](#) : une chaîne qui ne contient aucun caractère U+002E FULL STOP (.) ni aucun caractère U+003A COLON (:), utilisée comme nom de propriété d'élément propriétaire (c'est-à-dire utilisé par l'auteur pour des à des fins non définies dans une spécification publique).

Les spécifications qui introduisent [des noms de propriété définis](#) doivent garantir que tous ces noms de propriété ne contiennent aucun caractère U+002E FULL STOP (.), aucun caractère U+003A COLON (:) et aucun [espace ASCII](#) .

*Les règles ci-dessus interdisent les caractères U+003A COLON (:) dans les valeurs autres que les URL, car sinon, ils ne pourraient pas être distingués des URL. Les*

valeurs avec U+002E FULL STOP (.) sont réservées pour de futures extensions. [Les espaces blancs ASCII](#) ne sont pas autorisés car sinon les valeurs seraient analysées comme plusieurs jetons.

Lorsqu'un élément avec un [itemprop](#) attribut [ajoute une propriété](#) à plusieurs [éléments](#), l'exigence ci-dessus concernant les jetons s'applique à chaque [élément](#) individuellement.

Les **noms de propriété** d'un élément sont les jetons que l' [itemprop](#) attribut de l'élément contient lorsque sa valeur est [fractionnée sur ASCII whitespace](#), avec l'ordre conservé mais avec les doublons supprimés (ne laissant que la première occurrence de chaque nom).

Au sein d'un [élément](#), les propriétés ne sont pas ordonnées les unes par rapport aux autres, à l'exception des propriétés portant le même nom, qui sont ordonnées dans l'ordre dans lequel elles sont données par l'algorithme qui définit [les propriétés d'un élément](#).

Dans l'exemple suivant, la propriété "a" a les valeurs "1" et "2", *dans cet ordre*, mais que la propriété "a" soit placée avant la propriété "b" ou non n'a pas d'importance :

```
<div itemscope>
  <p itemprop="a">1</p>
  <p itemprop="a">2</p>
  <p itemprop="b">test</p>
</div>
```

Ainsi, ce qui suit est équivalent :

```
<div itemscope>
  <p itemprop="b">test</p>
  <p itemprop="a">1</p>
  <p itemprop="a">2</p>
</div>
```

Comme c'est le suivant :

```
<div itemscope>
  <p itemprop="a">1</p>
  <p itemprop="b">test</p>
  <p itemprop="a">2</p>
</div>
```

Et les suivants :

```
<div id="x">
  <p itemprop="a">1</p>
```



```
</div>
<div itemscope itemref="x">
  <p itemprop="b">test</p>
  <p itemprop="a">2</p>
</div>
```

## 5.2.4 Valeurs

La **valeur de propriété** d'une paire nom-valeur ajoutée par un élément avec un itemprop attribut est celle donnée pour le premier cas correspondant dans la liste suivante :

**Si l'élément a aussi un itemscope attribut**

La valeur est l' élément créé par l'élément.

**Si l'élément est un meta élément**

La valeur est la valeur de l' content attribut de l'élément, le cas échéant, ou la chaîne vide s'il n'y a pas un tel attribut.

**Si l'élément est un élément , audio, embed, iframe, , ou imgsrcsettrackvideo**

La valeur est la chaîne URL résultante qui résulte de l'analyse de la valeur de l' src attribut de l'élément par rapport au document de nœud de l'élément au moment où l'attribut est défini, ou la chaîne vide s'il n'y a pas un tel attribut ou si l' analyse aboutit à un erreur.

**Si l'élément est un élément a, area ou link**

La valeur est la chaîne URL résultante qui résulte de l'analyse de la valeur de l' href attribut de l'élément par rapport au document de nœud de l'élément au moment où l'attribut est défini, ou la chaîne vide s'il n'y a pas un tel attribut ou si l' analyse aboutit à un erreur.

**Si l'élément est un object élément**

La valeur est la chaîne URL résultante qui résulte de l'analyse de la valeur de l' data attribut de l'élément par rapport au document de nœud de l'élément au moment où l'attribut est défini, ou la chaîne vide s'il n'y a pas un tel attribut ou si l' analyse aboutit à un erreur.

**Si l'élément est un data élément**

La valeur est la valeur de l' value attribut de l'élément, s'il en a un, ou la chaîne vide sinon.

**Si l'élément est un meter élément**

La valeur est la valeur de l' valueattribut de l'élément, s'il en a un, ou la chaîne vide sinon.

#### Si l'élément est un timeélément

La valeur est la valeur datetime de l'élément .

#### Sinon

La valeur est le contenu textuel descendant de l'élément .

Les **éléments de propriété d'URL** sont les éléments a, area, audio, embed, iframe, img, link, , , et objectsourcetrackvideo

Si la valeur d'une propriété , telle que définie par la définition de la propriété, est une URL absolue , la propriété doit être spécifiée à l'aide d'un élément de propriété d'URL .

*Ces exigences ne s'appliquent pas simplement parce qu'une valeur de propriété correspond à la syntaxe d'une URL. Ils ne s'appliquent que si la propriété est explicitement définie comme prenant une telle valeur.*

Par exemple, un livre sur le premier alunissage pourrait s'appeler "mission:moon". Une propriété "title" d'un vocabulaire qui définit un titre comme étant une chaîne ne s'attendrait pas à ce que le titre soit donné dans un aélément, même s'il ressemble à une URL . D'un autre côté, s'il y avait un vocabulaire (plutôt limité !) pour les "livres dont les titres ressemblent à des URL" qui avaient une propriété "title" définie pour prendre une URL, alors la propriété s'attendrait à ce que le titre soit donné *dans* un aelement (ou l'un des autres éléments de propriété d'URL ), en raison de l'exigence ci-dessus.

### 5.2.5 Associer des noms à des éléments

Pour trouver **les propriétés d'un élément** défini par l'élément *root* , l'agent utilisateur doit exécuter les étapes suivantes. Ces étapes sont également utilisées pour signaler les erreurs de microdonnées .

1. Soit *results* , *memory* et *pending* des listes vides d'éléments.
2. Ajoutez la *racine* de l'élément à *la mémoire* .
3. Ajoutez les éléments enfants de *root* , le cas échéant, à *pending* .
4. Si *root* a un itemrefattribut, divisez la valeur de cet itemrefattribut sur l'espace blanc ASCII . Pour chaque *ID* de jeton résultant , s'il existe un élément dans l' arborescence de *root* avec l' ID *ID* , ajoutez le premier élément de ce type à *pending* .

5. Tant que *l'attente* n'est pas vide :

1. Supprimez un élément de *l'attente* et laissez *courant* être cet élément.
  2. Si *le courant* est déjà en *mémoire* , il y a une [erreur de microdonnées](#) ; [continuer](#) .
  3. Ajoutez *du courant* à *la mémoire* .
  4. Si *current* n'a pas d' [itemscope](#) attribut, alors : ajoutez tous les éléments enfants de *current* à *pending* .
  5. Si *courant* a un [itemprop](#) attribut spécifié et a un ou plusieurs [noms de propriété](#) , alors ajoutez *courant* aux *résultats* .
6. Trier *les résultats* dans [l'ordre de l'arborescence](#) .
7. Renvoyer *les résultats* .

Un document ne doit pas contenir d' [éléments](#) pour lesquels l'algorithme de recherche [des propriétés d'un élément](#) détecte des **erreurs de microdonnées** .

Un [élément](#) est un **élément de microdonnées de niveau supérieur** si son élément n'a pas d' [itemprop](#) attribut.

Tous [itemref](#) les attributs de a [Document](#) doivent être tels qu'il n'y ait pas de cycles dans le graphe formé à partir de la représentation de chaque [élément](#) de sous la [Document](#) forme d'un nœud dans le graphe et de chaque [propriété](#) d'un élément dont [la valeur](#) est un autre élément sous la forme d'une arête du graphe reliant ces deux éléments.

Un document ne doit pas contenir d'éléments ayant un [itemprop](#) attribut qui ne serait pas considéré comme une propriété de l'un des [éléments](#) de ce document si leurs [propriétés](#) devaient toutes être déterminées.

Dans cet exemple, une seule déclaration de licence est appliquée à deux œuvres, en utilisant [itemref](#) parmi les éléments représentant les œuvres :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>Photo gallery</title>
  </head>
  <body>
    <h1>My photos</h1>
    <figure itemscope itemtype="http://n.whatwg.org/work"
itemref="licenses">
      
```

```

    <figcaption itemprop="title">The house I found.</figcaption>
  </figure>

  <figure itemscope itemtype="http://n.whatwg.org/work"
    itemref="licenses">
    

    <figcaption itemprop="title">The mailbox.</figcaption>
  </figure>

  <footer>
    <p id="licenses">All images licensed under the <a
    itemprop="license"
      href="http://www.opensource.org/licenses/mit-license.php">MIT
      license</a>.</p>
  </footer>
</body>
</html>

```

Les résultats ci-dessus donnent deux éléments avec le type  
 " http://n.whatwg.org/work", l'un avec :

#### travail

images/house.jpeg

#### titre

La maison que j'ai trouvée.

#### Licence

http://www.opensource.org/licenses/mit-license.php

...et une avec :

#### travail

images/mailbox.jpeg

#### titre

La boîte aux lettres.

#### Licence

http://www.opensource.org/licenses/mit-license.php

## 5.2.6 Microdonnées et autres espaces de noms

Actuellement, les itemscope attributs itemprop, et autres microdonnées ne sont  
 définis que pour les éléments HTML . Cela signifie que les attributs avec les noms

littéraux " " `itemscope`, " `itemprop`", etc., n'entraînent pas le traitement des microdonnées sur les éléments dans d'autres espaces de noms, tels que SVG.

Ainsi, dans l'exemple suivant, il n'y a qu'un seul élément, pas deux.

```
<p itemscope></p> <!-- this is an item (with no properties and no type) -->

<svg itemscope></svg> <!-- this is not, it's just an SVG svg element with an invalid unknown attribute -->
```

## 5.3 Exemples de vocabulaires de microdonnées

Les vocabulaires de cette section sont principalement destinés à démontrer comment un vocabulaire est spécifié, bien qu'ils soient également utilisables en tant que tels.

### 5.3.1 vCard

Un élément avec le [type d'élément](#) <http://microformats.org/profile/hcard> représente les informations de contact d'une personne ou d'une organisation.

Ce vocabulaire ne prend pas [en charge les identificateurs globaux pour les éléments](#) .

Voici les [noms de propriété définis](#) du type . Ils sont basés sur le vocabulaire défini dans *vCard Format Specification* ( *vCard* ) et ses extensions, où plus d'informations sur la façon d'interpréter les valeurs peuvent être trouvées. [\[RFC6350\]](#)

#### **kind**

Décrit le type de contact représenté par l'élément.

La [valeur](#) doit être un texte identique à l'une des [chaînes kind](#) .

Une seule propriété avec le nom [kind](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcard>.

#### **fn**

Donne le texte formaté correspondant au nom de la personne ou de l'organisation.

La [valeur](#) doit être du texte.

Exactement une propriété avec le nom fn doit être présente dans chaque élément avec le type <http://microformats.org/profile/hcard>.

**n**

Donne le nom structuré de la personne ou de l'organisation.

La valeur doit être un élément avec zéro ou plus de chacune des propriétés family-name, given-name, additional-name, honorific-prefix et honorific-suffix.

Exactement une propriété avec le nom n doit être présente dans chaque élément avec le type <http://microformats.org/profile/hcard>.

**family-name**(à l'intérieur n)

Indique le nom de famille de la personne ou le nom complet de l'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom family-name peut être présent dans l' élément qui forme la valeur de la n propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

**given-name**(à l'intérieur n)

Donne le prénom de la personne.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom given-name peut être présent dans l' élément qui forme la valeur de la n propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

**additional-name**(à l'intérieur n)

Donne les noms supplémentaires de la personne.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom additional-name peut être présent dans l' élément qui forme la valeur de la n propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

**honorific-prefix**(à l'intérieur n)

Donne le préfixe honorifique de la personne.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom honorific-prefix peut être présent dans l' élément qui forme la valeur de la npropriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **honorific-suffix(à l'intérieur n)**

Donne le suffixe honorifique de la personne.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom honorific-suffix peut être présent dans l' élément qui forme la valeur de la npropriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **nickname**

Donne le surnom de la personne ou de l'organisation.

*Le surnom est le nom descriptif donné à la place ou en plus de celui appartenant à une personne, un lieu ou une chose. Il peut également être utilisé pour spécifier une forme familière d'un nom propre spécifié par les propriétés fn ou n.*

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom nickname peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **photo**

Donne une photographie de la personne ou de l'organisation.

La valeur doit être une URL absolue .

N'importe quel nombre de propriétés avec le nom photo peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **bday**

Donne la date de naissance de la personne ou de l'organisation.

La valeur doit être une chaîne de date valide .

Une seule propriété avec le nom bday peut être présente dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **anniversary**

Donne la date de naissance de la personne ou de l'organisation.

La valeur doit être une chaîne de date valide .

Une seule propriété avec le nom anniversary peut être présente dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **sex**

Donne le sexe biologique de la personne.

La valeur doit être l'une des suivantes F : , signifiant « féminin », M, signifiant « masculin », N, signifiant « aucun ou sans objet », O, signifiant « autre » ou U, signifiant « inconnu ».

Une seule propriété avec le nom sex peut être présente dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **gender-identity**

Donne l'identité de genre de la personne.

La valeur doit être du texte.

Une seule propriété avec le nom gender-identity peut être présente dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **adr**

Donne l'adresse de livraison de la personne ou de l'organisation.

La valeur doit être un élément avec zéro ou plusieurs propriétés type, post-office-box, extended-address et street-address, et éventuellement une locality propriété, éventuellement une region propriété, éventuellement une postal-code propriété et éventuellement une country-name propriété.

Si aucune type propriété n'est présente dans un élément qui forme la valeur d'une adr propriété d'un élément avec le type <http://microformats.org/profile/hcard>, la chaîne de type d'adresse work est implicite.

N'importe quel nombre de propriétés avec le nom adr peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **type(à l'intérieur adr)**

Indique le type d'adresse de livraison.

La valeur doit être un texte identique à l'une des chaînes de type d'adresse .



N'importe quel nombre de propriétés avec le nom type peut être présent dans l' élément qui forme la valeur d'une adr propriété d'un élément avec le type <http://microformats.org/profile/hcard>, mais dans chaque élémentadr de propriété de ce type , il ne doit y avoir qu'une seule propriété par valeur distincte.type

**post-office-box(à l'intérieur adr)**

Indique la case postale de l'adresse de livraison de la personne ou de l'organisme.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom post-office-box peut être présent dans l' élément qui forme la valeur d'une adr propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

*vCard invite les auteurs à ne pas utiliser ce champ.*

**extended-address(à l'intérieur adr)**

Donne un élément supplémentaire de l'adresse de livraison de la personne ou de l'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom extended-address peut être présent dans l' élément qui forme la valeur d'une adr propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

*vCard invite les auteurs à ne pas utiliser ce champ.*

**street-address(à l'intérieur adr)**

Donne la composante de l'adresse municipale de l'adresse de livraison de la personne ou de l'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom street-address peut être présent dans l' élément qui forme la valeur d'une adr propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

**locality(à l'intérieur adr)**

Donne la composante de localité (par exemple la ville) de l'adresse de livraison de la personne ou de l'organisation.

La valeur doit être du texte.

Une seule propriété avec le nom locality peut être présente dans l' élément qui forme la valeur d'une adrpropriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **region**(à l'intérieur adr)

Donne la composante régionale (par exemple état ou province) de l'adresse de livraison de la personne ou de l'organisation.

La valeur doit être du texte.

Une seule propriété avec le nom region peut être présente dans l' élément qui forme la valeur d'une adrpropriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **postal-code**(à l'intérieur adr)

Donne la composante code postal de l'adresse de livraison de la personne ou de l'organisation.

La valeur doit être du texte.

Une seule propriété avec le nom postal-code peut être présente dans l' élément qui forme la valeur d'une adrpropriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **country-name**(à l'intérieur adr)

Donne le nom du pays composant l'adresse de livraison de la personne ou de l'organisation.

La valeur doit être du texte.

Une seule propriété avec le nom country-name peut être présente dans l' élément qui forme la valeur d'une adrpropriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **tel**

Indique le numéro de téléphone de la personne ou de l'organisme.

La valeur doit être soit du texte pouvant être interprété comme un numéro de téléphone tel que défini dans les spécifications CCITT E.163 et X.121, soit un élément avec zéro ou plusieurs typepropriétés et exactement une valuepropriété. [\[E163\]](#) [\[X121\]](#)

Si aucune typepropriété n'est présente dans un élément qui forme la valeur d'une telpropriété d'un élément avec le type <http://microformats.org/profile/hcard>, ou si la valeur d'une telle tel propriété est du texte, alors la chaîne de type de téléphone [voice](#) est implicite.

N'importe quel nombre de propriétés avec le nom tel peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **type**(à l'intérieur tel)

Indique le type de numéro de téléphone.

La valeur doit être un texte identique à l'une des chaînes de type de téléphone .

N'importe quel nombre de propriétés avec le nom type peut être présent dans l' élément qui forme la valeur d'une tel propriété d'un élément avec le type <http://microformats.org/profile/hcard>, mais dans chaque élémenttel de propriété de ce type , il ne doit y avoir qu'une seule propriété par valeur distincte.type

#### **value**(à l'intérieur tel)

Donne le numéro de téléphone réel de la personne ou de l'organisation.

La valeur doit être un texte pouvant être interprété comme un numéro de téléphone tel que défini dans les spécifications CCITT E.163 et X.121. [\[E163\]](#) [\[X121\]](#)

Exactement une propriété avec le nom value doit être présente dans l' élément qui forme la valeur d'une tel propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **email**

Indique l'adresse e-mail de la personne ou de l'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom email peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **impp**

Fournit une URL pour la messagerie instantanée et les communications de protocole de présence avec la personne ou l'organisation.

La valeur doit être une URL absolue .

N'importe quel nombre de propriétés avec le nom impp peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **lang**

Donne une langue comprise par la personne ou l'organisation.

La [valeur](#) doit être une balise de langue BCP 47 valide. [\[BCP47\]](#) .

N'importe quel nombre de propriétés avec le nom [lang](#) peut être présent dans chaque [élément](#) avec le type <http://microformats.org/profile/hcard>.

#### tz

Indique le fuseau horaire de la personne ou de l'organisation.

La [valeur](#) doit être du texte et doit correspondre à la syntaxe suivante :

1. Soit un caractère U+002B SIGNE PLUS (+) ou un caractère U+002D TIRET MOINS (-).
2. Un [entier non négatif valide](#) qui comporte exactement deux chiffres et qui représente un nombre compris entre 00 et 23.
3. Un caractère U+003A COLON (:).
4. Un [entier non négatif valide](#) qui comporte exactement deux chiffres et qui représente un nombre compris entre 00 et 59.

N'importe quel nombre de propriétés avec le nom [tz](#) peut être présent dans chaque [élément](#) avec le type <http://microformats.org/profile/hcard>.

#### geo

Donne la position géographique de la personne ou de l'organisation.

La [valeur](#) doit être du texte et doit correspondre à la syntaxe suivante :

1. Facultativement, un caractère U+002B SIGNE PLUS (+) ou un caractère U+002D TIRET MOINS (-).
2. Un ou plusieurs [chiffres ASCII](#) .
3. Facultativement\*, un caractère U+002E FULL STOP (.) suivi d'un ou plusieurs [chiffres ASCII](#) .
4. Un caractère POINT-virgule U+003B (;).
5. Facultativement, un caractère U+002B SIGNE PLUS (+) ou un caractère U+002D TIRET MOINS (-).
6. Un ou plusieurs [chiffres ASCII](#) .
7. Facultativement\*, un caractère U+002E FULL STOP (.) suivi d'un ou plusieurs [chiffres ASCII](#) .

Les composants facultatifs marqués d'un astérisque (\*) doivent être inclus et doivent comporter six chiffres chacun.

*La valeur spécifie la latitude et la longitude, dans cet ordre (c'est-à-dire, l'ordre "LAT LON"), en degrés décimaux. La longitude représente l'emplacement à l'est et à l'ouest du premier méridien sous la forme d'un nombre réel positif ou négatif, respectivement. La latitude représente l'emplacement au nord et au sud de l'équateur sous la forme d'un nombre réel positif ou négatif, respectivement.*

N'importe quel nombre de propriétés avec le nom geo peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### title

Donne le titre du poste, le poste fonctionnel ou la fonction de la personne ou de l'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom title peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### role

Indique le rôle, la profession ou la catégorie d'activité de la personne ou de l'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom role peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### logo

Donne le logo de la personne ou de l'organisation.

La valeur doit être une URL absolue .

N'importe quel nombre de propriétés avec le nom logo peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### agent

Donne les coordonnées d'une autre personne qui agira au nom de la personne ou de l'organisation.

La valeur doit être soit un élément avec le type <http://microformats.org/profile/hcard>, soit une URL absolue, soit du texte.

N'importe quel nombre de propriétés avec le nom agent peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **org**

Donne le nom et les unités de l'organisation.

La valeur doit être soit du texte, soit un élément avec une organization-name propriété et zéro ou plusieurs organization-unit propriétés.

N'importe quel nombre de propriétés avec le nom org peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **organization-name(à l'intérieur org)**

Donne le nom de l'organisation.

La valeur doit être du texte.

Exactement une propriété avec le nom organization-name doit être présente dans l' élément qui forme la valeur d'une org propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **organization-unit(à l'intérieur org)**

Donne le nom de l'unité d'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom organization-unit peut être présent dans l' élément qui forme la valeur de la org propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **member**

Donne une URL qui représente un membre du groupe.

La valeur doit être une URL absolue.

N'importe quel nombre de propriétés avec le nom member peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard> si l' élément a également une propriété avec le nom kind dont la valeur est "group".

#### **related**

Donne une relation à une autre entité.

La valeur doit être un élément avec une url propriété et une rel propriété.

N'importe quel nombre de propriétés avec le nom related peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **url**(à l'intérieur related)

Donne l' URL de l'entité associée.

La valeur doit être une URL absolue .

Exactement une propriété avec le nom url doit être présente dans l' élément qui forme la valeur d'une related propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **rel**(à l'intérieur related)

Donne la relation entre l'entité et l'entité associée.

La valeur doit être un texte identique à l'une des chaînes de relation .

Exactement une propriété avec le nom rel doit être présente dans l' élément qui forme la valeur d'une related propriété d'un élément avec le type <http://microformats.org/profile/hcard>.

#### **categories**

Donne le nom d'une catégorie ou d'un tag dans lequel la personne ou l'organisation pourrait être classée.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom categories peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **note**

Donne des informations supplémentaires ou un commentaire sur la personne ou l'organisation.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom note peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **rev**

Donne la date et l'heure de révision des coordonnées.

La valeur doit être un texte correspondant à une chaîne de date et d'heure globale valide .

*La valeur distingue la révision actuelle des informations des autres rendus des informations.*

N'importe quel nombre de propriétés avec le nom rev peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **sound**

Donne un fichier sonore relatif à la personne ou à l'organisation.

La valeur doit être une URL absolue .

N'importe quel nombre de propriétés avec le nom sound peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **uid**

Donne un identifiant global unique correspondant à la personne ou à l'organisation.

La valeur doit être du texte.

Une seule propriété avec le nom uid peut être présente dans chaque élément avec le type <http://microformats.org/profile/hcard>.

#### **url**

Donne une URL relative à la personne ou à l'organisation.

La valeur doit être une URL absolue .

N'importe quel nombre de propriétés avec le nom url peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcard>.

**Les chaînes de type sont :**

#### **individual**

Indique une seule entité (par exemple une personne).

#### **group**

Indique plusieurs entités (par exemple une liste de diffusion).

#### **org**



Indique une seule entité qui n'est pas une personne (par exemple une entreprise).

**location**

Indique un lieu géographique (par exemple un immeuble de bureaux).

**Les chaînes de type d'adresse sont :**

**home**

Indique une adresse de livraison pour une résidence.

**work**

Indique une adresse de livraison pour un lieu de travail.

**Les chaînes de type de téléphone sont :**

**home**

Indique un numéro résidentiel.

**work**

Indique un numéro de téléphone pour un lieu de travail.

**text**

Indique que le numéro de téléphone prend en charge les messages texte (SMS).

**voice**

Indique un numéro de téléphone vocal.

**fax**

Indique un numéro de télécopieur.

**cell**

Indique un numéro de téléphone cellulaire.

**video**

Indique un numéro de téléphone de visioconférence.

**pager**

Indique le numéro de téléphone d'un appareil de radiomessagerie.

**textphone**

Indique un appareil de télécommunication pour les personnes ayant des difficultés d'audition ou d'élocution.

**Les chaînes de relation sont :**

**emergency**

Un contact d'urgence.

**agent**

Une autre entité qui agit au nom de cette entité.

**contact**

**connaissance**

**ami**

**rencontré**

**ouvrier**

**collègue**

**résident**

**voisin**

**enfant**

**parent**

**frère et sœur**

**conjoint**

**proche**

**muse**

**écraser**

**date**

**amour**

**moi**

A la signification définie dans XFN. [\[XFN\]](#)

### 5.3.1.1 Conversion en vCard

Étant donné une liste de nœuds *nodes* dans un [Document](#), un agent utilisateur doit exécuter l'algorithme suivant pour **extraire toutes les données vCard représentées par ces nœuds** (seule la première vCard est renvoyée) :

1. Si aucun des nœuds dans *nodes* n'est [un élément](#) avec le [type d'élément](#) <http://microformats.org/profile/hcard>, alors il n'y a pas de vCard. Abandonne l'algorithme, ne renvoyant rien.
2. Soit *node* le premier nœud dans *nodes* qui est un [élément](#) avec le [type d'élément](#) <http://microformats.org/profile/hcard>.
3. Soit *la sortie* une chaîne vide.
4. [Ajoutez une ligne vCard](#) avec le type " BEGIN" et la valeur " vCARD" à *la sortie*.

5. [Ajoutez une ligne vCard](#) avec le type " `PROFILE`" et la valeur " `VCARD`" à la sortie .
6. [Ajoutez une ligne vCard](#) avec le type " `VERSION`" et la valeur " `4.0`" à la sortie .
7. [Ajoutez une ligne vCard](#) avec le type " `SOURCE`" et le résultat de [l'échappement de la chaîne de texte vCard](#) qui est l' [URL](#) du document comme valeur à sortir .
8. Si l' [title](#)élément n'est pas nul, [ajoutez une ligne vCard](#) avec le type " `NAME`" et avec le résultat de [l'échappement de la chaîne de texte vCard](#) obtenue à partir [du titlecontenu textuel descendant](#) de l' [élément](#) comme valeur à afficher .
9. Que *le sexe* soit la chaîne vide.
10. Soit *gender-identity* la chaîne vide.
11. Pour chaque élément *element* qui est [une propriété du nœud](#) d'élément : pour chaque nom *name* dans [les noms de propriété](#) de l'élément , exécutez les sous-étapes suivantes :

1. Soit *parameters* un ensemble vide de paires nom-valeur.
2. Exécutez l'ensemble approprié de sous-étapes dans la liste suivante. Les étapes définiront une *valeur* de variable , qui sera utilisée à l'étape suivante.

**Si la [valeur](#) de la propriété est un *sous-élément d' élément* et que le nom est [n](#)**

1. Soit *value* la chaîne vide.
2. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [family-name](#) dans *subitem* .
3. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .
4. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [given-name](#) dans *subitem* .
5. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .
6. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [additional-name](#) dans *subitem* .
7. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .
8. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [honorific-prefix](#) dans *subitem* .
9. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .

10. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [honorific-suffix](#) dans *subitem* .

**Si la [valeur](#) de la propriété est un sous-élément d' [élément](#) et que le nom est [adr](#)**

11. Soit *value* la chaîne vide.

12. Ajouter à *la valeur* le résultat de [la collecte des sous-propriétés vCard](#) nommées [post-office-box](#) dans *subitem* .

13. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .

14. Ajouter à *la valeur* le résultat de [la collecte des sous-propriétés vCard](#) nommées [extended-address](#) dans *subitem* .

15. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .

16. Ajouter à *la valeur* le résultat de [la collecte des sous-propriétés vCard](#) nommées [street-address](#) dans *subitem* .

17. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .

18. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [locality](#) dans *subitem* .

19. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .

20. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [region](#) dans *subitem* .

21. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .

22. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [postal-code](#) dans *subitem* .

23. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .

24. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [country-name](#) dans *subitem* .

25. S'il existe une propriété nommée [type](#) dans *subitem* , et que la première de ces propriétés a une [valeur](#) qui n'est pas un [élément](#) et dont la valeur se compose uniquement de [caractères alphanumériques ASCII](#) , alors ajoutez un paramètre nommé " TYPE " dont la valeur est la [valeur](#) de cette propriété à *parameters* .

**Si la [valeur](#) de la propriété est un sous-élément d' [élément](#) et que le nom est [org](#)**

26. Soit *value* la chaîne vide.

27. Ajouter à *la valeur* le résultat de [la collecte de la première sous-propriété vCard](#) nommée [organization-name](#) dans *subitem* .
28. Pour chaque propriété nommée [organization-unit](#) dans *subitem* , exécutez les étapes suivantes :
1. Si la [valeur](#) de la propriété est un [élément](#) , ignorez cette propriété.
  2. Ajoutez un caractère U+003B SEMICOLON (;) à *la valeur* .
  3. Ajoutez le résultat de [l'échappement de la chaîne de texte vCard](#) donnée par la [valeur](#) de la propriété à *value* .

**Si la [valeur](#) de la propriété est un sous-élément [d'élément](#) avec le [type d'élément](#) et le nom est <http://microformats.org/profile/hcardrelated>**

29. Soit *value* la chaîne vide.
30. S'il existe une propriété nommée [url](#) dans *subitem* et que son élément est un [élément de propriété d'URL](#) , ajoutez le résultat de [l'échappement de la chaîne de texte vCard](#) donnée par la [valeur](#) de la première propriété de ce type à *value* et ajoutez un paramètre avec le nom " VALUE " et la valeur " URI " aux *paramètres* .
31. S'il existe une propriété nommée [rel](#) dans *subitem* , et que la première de ces propriétés a une [valeur](#) qui n'est pas un [élément](#) et dont la valeur se compose uniquement de [caractères alphanumériques ASCII](#) , alors ajoutez un paramètre nommé " RELATION " dont la valeur est la [valeur](#) de cette propriété à *parameters* .

**Si la [valeur](#) de la propriété est un [élément](#) et que le nom n'est aucun des éléments ci-dessus**

32. Soit *value* le résultat de [la collecte de la première sous-propriété vCard](#) nommée *value* dans *subitem* .
33. S'il existe une propriété nommée *type* dans *subitem* , et que la première de ces propriétés a une [valeur](#) qui n'est pas un [élément](#) et dont la valeur consiste uniquement en [ASCII alphanumérique](#) , alors ajoutez un paramètre nommé " TYPE " dont la valeur est la [valeur](#) de cette propriété à *parameters* .

**Si la [valeur](#) de la propriété n'est pas un [élément](#) et que son nom est [sex](#)**

S'il s'agit de la première propriété de ce type à être trouvée, définissez *sex* sur la [valeur](#) de la propriété .

**Si la [valeur](#) de la propriété n'est pas un [élément](#) et que son nom est [gender-identity](#)**

S'il s'agit de la première propriété de ce type à être trouvée, définissez *gender-identity* sur la [valeur](#) de la propriété .

**Sinon (la [valeur](#) de la propriété n'est pas un [élément](#) )**

34. Soit *value* la [valeur](#) de la propriété .
35. Si l' *élément* est l' un des [éléments de propriété de l' URL](#) , ajoutez un paramètre avec le nom " `VALUE`" et la valeur " `URI`" aux *paramètres* .
36. Sinon, si le *nom* est [bday](#) ou [anniversary](#) et que la *valeur* est une [chaîne de date valide](#) , ajoutez un paramètre avec le nom " `VALUE`" et la valeur " `DATE`" à *parameters* .
37. Sinon, si *name* est [rev](#) et que la *valeur* est une [chaîne de date et d'heure globale valide](#) , ajoutez un paramètre avec le nom " `VALUE`" et la valeur " `DATE-TIME`" à *parameters* .
38. Préfixez chaque caractère U+005C REVERSE SOLIDUS (\) dans la *valeur* avec un autre caractère U+005C REVERSE SOLIDUS (\).
39. Préfixez chaque caractère virgule U+002C (,) dans la *valeur* avec un caractère U+005C REVERSE SOLIDUS (\).
40. Sauf si *name* est [geo](#), préfixez chaque caractère U+003B SEMICOLON (;) dans *value* avec un caractère U+005C REVERSE SOLIDUS (\).
41. Remplacez chaque paire de caractères U+000D CARRIAGE RETURN U+000A LINE FEED (CRLF) dans la *valeur* par un caractère U+005C REVERSE SOLIDUS (\) suivi d'un caractère U+006E LATIN SMALL LETTER N (n).
42. Remplacez chaque caractère U+000D CARRIAGE RETURN (CR) ou U+000A LINE FEED (LF) restant dans la *valeur* par un caractère U+005C REVERSE SOLIDUS (\) suivi d'un caractère U+006E LATIN SMALL LETTER N (n).
3. [Ajoutez une ligne vCard](#) avec le *nom* du type , les *paramètres* *parameters* et la *valeur* *value* à *sortir* .
12. Si *sex* ou *gender-identity* a une valeur qui n'est pas la chaîne vide, [ajoutez une ligne vCard](#) avec le type " `GENDER`" et la valeur consistant en la concaténation de *sex* , un caractère U+003B SEMICOLON (;) et *gender-identity* à la *sortie* .
13. [Ajoutez une ligne vCard](#) avec le type " `END`" et la valeur " `VCARD`" à la *sortie* .

Lorsque l'algorithme ci-dessus indique que l'agent utilisateur doit **ajouter une ligne vCard** composée d'un type *type* , éventuellement de certains paramètres et d'une valeur *value* à une chaîne *output* , il doit exécuter les étapes suivantes :

1. Soit *ligne* une chaîne vide.
2. Ajouter le type , [converti en majuscule ASCII](#) , à *la ligne* .
3. S'il y a des paramètres, alors pour chaque paramètre, dans l'ordre dans lequel ils ont été ajoutés, exécutez ces sous-étapes :
  1. Ajoutez un caractère U+003B SEMICOLON (;) à *la ligne* .
  2. Ajoutez le nom du paramètre à *la ligne* .
  3. Ajoutez un caractère SIGNE ÉGAL U+003D (=) à *la ligne* .
  4. Ajoutez la valeur du paramètre à *la ligne* .
4. Ajoutez un caractère U+003A COLON (:) à *la ligne* .
5. Ajouter *la valeur* à *la ligne* .
6. Laissez *la longueur maximale* être 75.
7. Tant que [la longueur du point de code](#) de *la ligne* est supérieure à *la longueur maximale* :
  1. Ajoutez les premiers points de code de *longueur maximale de la ligne* à *la sortie* .
  2. Supprimez les premiers points de code de *longueur maximale de la ligne* .
  3. Ajoutez un caractère U+000D RETOUR CHARIOT (CR) à *la sortie* .
  4. Ajoutez un caractère LINE FEED U+000A (LF) à *la sortie* .
  5. Ajoutez un caractère U+0020 SPACE à *la sortie* .
  6. Laissez *la longueur maximale* être 74.
8. Ajouter (ce qui reste de) *la ligne* à *la sortie* .
9. Ajoutez un caractère U+000D RETOUR CHARIOT (CR) à *la sortie* .
10. Ajoutez un caractère LINE FEED U+000A (LF) à *la sortie* .

Lorsque les étapes ci-dessus nécessitent que l'agent utilisateur obtienne le résultat de **la collecte des sous-propriétés vCard** nommées *subname* dans *subitem* , l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit *value* la chaîne vide.
2. Pour chaque propriété nommée *subname* dans l'élément *subitem* , exécutez les sous-étapes suivantes :

1. Si la valeur de la propriété est elle-même un élément , ignorez cette propriété.
2. S'il ne s'agit pas de la première propriété nommée *sous-nom* dans *sous-élément* (en ignorant celles qui ont été ignorées à l'étape précédente), ajoutez un caractère virgule U+002C (,) à la *valeur* .
3. Ajoutez le résultat de l'échappement de la chaîne de texte vCard donnée par la valeur de la propriété à *value* .

3. *Valeur* de retour .

Lorsque les étapes ci-dessus nécessitent que l'agent utilisateur obtienne le résultat de **la collecte de la première sous-propriété vCard** nommée *subname* dans *subitem* , l'agent utilisateur doit exécuter les étapes suivantes :

1. S'il n'y a pas de propriétés nommées *subname* dans *subite* , renvoyez la chaîne vide.
2. Si la valeur de la première propriété nommée *sous-nom* dans *sous-élément* est un élément , renvoyez la chaîne vide.
3. Renvoie le résultat de l'échappement de la chaîne de texte vCard donnée par la valeur de la première propriété nommée *subname* dans *subitem* .

Lorsque les algorithmes ci-dessus indiquent que l'agent utilisateur doit échapper la **valeur de la chaîne de texte vCard** , l'agent utilisateur doit suivre les étapes suivantes :

1. Préfixez chaque caractère U+005C REVERSE SOLIDUS (\) dans la *valeur* avec un autre caractère U+005C REVERSE SOLIDUS (\).
2. Préfixez chaque caractère virgule U+002C (,) dans la *valeur* avec un caractère U+005C REVERSE SOLIDUS (\).
3. Préfixez chaque caractère U+003B SEMICOLON (;) dans la *valeur* avec un caractère U+005C REVERSE SOLIDUS (\).
4. Remplacez chaque paire de caractères U+000D CARRIAGE RETURN U+000A LINE FEED (CRLF) dans la *valeur* par un caractère U+005C REVERSE SOLIDUS (\) suivi d'un caractère U+006E LATIN SMALL LETTER N (n).
5. Remplacez chaque caractère U+000D CARRIAGE RETURN (CR) ou U+000A LINE FEED (LF) restant dans la *valeur* par un caractère U+005C REVERSE SOLIDUS (\) suivi d'un caractère U+006E LATIN SMALL LETTER N (n).
6. Renvoie la *valeur* mutée .



Cet algorithme peut générer une sortie vCard non valide, si l'entrée n'est pas conforme aux règles décrites pour le <http://microformats.org/profile/hcard> type d'élément et les noms de propriété définis .

### 5.3.1.2 Exemples

*Cette section est non normative.*

Voici un long exemple de vCard pour un personnage fictif appelé "Jack Bauer":

```
<section id="jack" itemscope
itemtype="http://microformats.org/profile/hcard">
  <h1 itemprop="fn">
    <span itemprop="n" itemscope>
      <span itemprop="given-name">Jack</span>
      <span itemprop="family-name">Bauer</span>
    </span>
  </h1>
  
  <p itemprop="org" itemscope>
    <span itemprop="organization-name">Counter-Terrorist Unit</span>
    (<span itemprop="organization-unit">Los Angeles Division</span>)
  </p>
  <p>
    <span itemprop="adr" itemscope>
      <span itemprop="street-address">10201 W. Pico Blvd.</span><br>
      <span itemprop="locality">Los Angeles</span>,
      <span itemprop="region">CA</span>
      <span itemprop="postal-code">90064</span><br>
      <span itemprop="country-name">United States</span><br>
    </span>
    <span itemprop="geo">34.052339;-118.410623</span>
  </p>
  <h2>Assorted Contact Methods</h2>
  <ul>
    <li itemprop="tel" itemscope>
      <span itemprop="value">+1 (310) 597 3781</span> <span
itemprop="type">work</span>
      <meta itemprop="type" content="voice">
    </li>
```

```

    <li><a itemprop="url"
href="https://en.wikipedia.org/wiki/Jack_Bauer">I'm on
Wikipedia</a>

    so you can leave a message on my user talk page.</li>

    <li><a itemprop="url" href="http://www.jackbauerfacts.com/">Jack
Bauer Facts</a></li>

    <li itemprop="email"><a
href="mailto:j.bauer@la.ctu.gov.invalid">j.bauer@la.ctu.gov.invalid
</a></li>

    <li itemprop="tel" itemscope>

        <span itemprop="value">+1 (310) 555 3781</span> <span>

        <meta itemprop="type" content="cell">mobile phone</span>

    </li>
</ul>

<ins datetime="2008-07-20 21:00:00+01:00">
    <meta itemprop="rev" content="2008-07-20 21:00:00+01:00">
    <p itemprop="tel" itemscope><strong>Update!</strong>
    My new <span itemprop="type">home</span> phone number is
    <span itemprop="value">01632 960 123</span>.</p>
</ins>
</section>

```

Le retour à la ligne impair est nécessaire car les retours à la ligne sont significatifs dans les microdonnées : les retours à la ligne seraient conservés lors d'une conversion au format vCard, par exemple.

Cet exemple montre les coordonnées d'un site (à l'aide de l' address élément) contenant une adresse avec deux composants de rue :

```

<address itemscope
itemtype="http://microformats.org/profile/hcard">
    <strong itemprop="fn"><span itemprop="n" itemscope><span
itemprop="given-name">Alfred</span>
    <span itemprop="family-name">Person</span></span></strong> <br>
    <span itemprop="adr" itemscope>
        <span itemprop="street-address">1600 Amphitheatre Parkway</span>
        <br>
        <span itemprop="street-address">Building 43, Second Floor</span>
        <br>
        <span itemprop="locality">Mountain View</span>,
        <span itemprop="region">CA</span> <span itemprop="postal-
code">94043</span>
    </span>
</address>

```

Le vocabulaire vCard peut être utilisé pour marquer simplement les noms des personnes :

```
<span itemscope itemtype="http://microformats.org/profile/hcard">
  ><span itemprop=fn><span itemprop="n" itemscope><span
    itemprop="given-name"
  >George</span> <span itemprop="family-name">Washington</span></span>
</span></span>
```

Cela crée un seul élément avec deux paires nom-valeur, l'une avec le nom "fn" et la valeur "George Washington", et l'autre avec le nom "n" et un deuxième élément comme valeur, le deuxième élément ayant le deux paires nom-valeur "prénom" et "nom de famille" avec les valeurs "George" et "Washington" respectivement. Ceci est défini pour mapper à la vCard suivante :

```
COMMENCER : VCARD
PROFIL : VCARD
VERSION : 4.0
SOURCE : adresse du document
FN : George Washington
N:Washington;Georges;;;
FIN:VCARD
```

### 5.3.2 vÉvénement

Un élément avec le [type d'élément](http://microformats.org/profile/hcalendar#vevent) <http://microformats.org/profile/hcalendar#vevent> représente un événement.

Ce vocabulaire ne prend pas [en charge les identificateurs globaux pour les éléments](#) .

Voici les [noms de propriété définis](#) du type . Ils sont basés sur le vocabulaire défini dans *Internet Calendaring and Scheduling Core Object Specification ( iCalendar )*, où plus d'informations sur la façon d'interpréter les valeurs peuvent être trouvées. [\[RFC5545\]](#)

*Seules les parties du vocabulaire iCalendar relatives aux événements sont utilisées ici ; ce vocabulaire ne peut pas exprimer une instance iCalendar complète.*

**attach**

Donne l'adresse d'un document associé à l'événement.

La [valeur](#) doit être une [URL absolue](#) .

N'importe quel nombre de propriétés avec le nom [attach](#) peut être présent dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

### categories

Donne le nom d'une catégorie ou d'un tag dans lequel l'événement pourrait être classé.

La [valeur](#) doit être du texte.

N'importe quel nombre de propriétés avec le nom `categories` peut être présent dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

### class

Donne la classification d'accès des informations relatives à l'événement.

La [valeur](#) doit être du texte avec l'une des valeurs suivantes :

- public
- private
- confidential

***Ceci est purement consultatif et ne peut être considéré comme une mesure de confidentialité.***

Une seule propriété avec le nom `class` peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

### comment

Donne un commentaire sur l'événement.

La [valeur](#) doit être du texte.

N'importe quel nombre de propriétés avec le nom `comment` peut être présent dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

### description

Donne une description détaillée de l'événement.

La [valeur](#) doit être du texte.

Une seule propriété avec le nom `description` peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

### geo

Donne la position géographique de l'événement.

La [valeur](#) doit être du texte et doit correspondre à la syntaxe suivante :

1. Facultativement, un caractère U+002B SIGNE PLUS (+) ou un caractère U+002D TIRET MOINS (-).
2. Un ou plusieurs [chiffres ASCII](#) .
3. Facultativement\*, un caractère U+002E FULL STOP (.) suivi d'un ou plusieurs [chiffres ASCII](#) .
4. Un caractère POINT-virgule U+003B (;).
5. Facultativement, un caractère U+002B SIGNE PLUS (+) ou un caractère U+002D TIRET MOINS (-).
6. Un ou plusieurs [chiffres ASCII](#) .
7. Facultativement\*, un caractère U+002E FULL STOP (.) suivi d'un ou plusieurs [chiffres ASCII](#) .

Les composants facultatifs marqués d'un astérisque (\*) doivent être inclus et doivent comporter six chiffres chacun.

*La valeur spécifie la latitude et la longitude, dans cet ordre (c'est-à-dire, l'ordre "LAT LON"), en degrés décimaux. La longitude représente l'emplacement à l'est et à l'ouest du premier méridien sous la forme d'un nombre réel positif ou négatif, respectivement. La latitude représente l'emplacement au nord et au sud de l'équateur sous la forme d'un nombre réel positif ou négatif, respectivement.*

Une seule propriété avec le nom [geo](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **location**

Donne le lieu de l'événement.

La [valeur](#) doit être du texte.

Une seule propriété avec le nom [location](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **resources**

Donne une ressource qui sera nécessaire pour l'événement.

La [valeur](#) doit être du texte.

N'importe quel nombre de propriétés avec le nom [resources](#) peut être présent dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **status**

Donne le statut de confirmation de l'événement.

La [valeur](#) doit être du texte avec l'une des valeurs suivantes :

- tentative
- confirmed
- canceled

Une seule propriété avec le nom [status](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### summary

Donne un bref résumé de l'événement.

La [valeur](#) doit être du texte.

Les agents utilisateurs doivent remplacer les caractères U+000A LINE FEED (LF) dans la [valeur](#) par les caractères U+0020 SPACE lors de l'utilisation de la valeur.

Une seule propriété avec le nom [summary](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### dtend

Donne la date et l'heure auxquelles l'événement se termine.

Si la propriété avec le nom [dtend](#) est présente dans un [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent> qui a une propriété avec le nom [dtstart](#) dont la valeur est une [chaîne de date valide](#), alors la [valeur](#) de la propriété avec le nom [dtend](#) doit être un texte qui est également une [chaîne de date valide](#). Sinon, la [valeur](#) de la propriété doit être un texte correspondant à une [chaîne de date et d'heure globale valide](#).

Dans les deux cas, la [valeur](#) doit être postérieure à la valeur de la [dtstart](#) propriété du même [élément](#).

*L'heure indiquée par l' [dtend](#) établissement n'est pas incluse. Pour les événements d'une journée, par conséquent, la [valeur](#) [dtend](#) de la propriété sera le lendemain de la fin de l'événement.*

Une seule propriété avec le nom [dtend](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>, tant qu'il <http://microformats.org/profile/hcalendar#vevent> n'a pas de propriété avec le nom [duration](#).

#### dtstart

Donne la date et l'heure de début de l'événement.

La valeur doit être du texte qui est soit une chaîne de date valide , soit une chaîne de date et d'heure globale valide .

Exactement une propriété avec le nom dtstart doit être présente dans chaque élément avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **duration**

Donne la durée de l'événement.

La valeur doit être un texte correspondant à une chaîne de durée d'événement valide .

La durée représentée est la somme de toutes les durées représentées par des entiers dans la valeur.

Une seule propriété avec le nom duration peut être présente dans chaque élément avec le type <http://microformats.org/profile/hcalendar#vevent>, tant qu'il <http://microformats.org/profile/hcalendar#vevent> n'a pas de propriété avec le nom dtend.

#### **transp**

Indique si l'événement est à considérer comme consommateur de temps sur un calendrier, dans le cadre des recherches de temps libre-occupé.

La valeur doit être du texte avec l'une des valeurs suivantes :

- opaque
- transparent

Une seule propriété avec le nom transp peut être présente dans chaque élément avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **contact**

Donne les coordonnées de l'événement.

La valeur doit être du texte.

N'importe quel nombre de propriétés avec le nom contact peut être présent dans chaque élément avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **url**

Donne une URL pour l'événement.

La valeur doit être une URL absolue .

Une seule propriété avec le nom `url` peut être présente dans chaque `élément` avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### `uid`

Donne un identifiant global unique correspondant à l'événement.

La `valeur` doit être du texte.

Une seule propriété avec le nom `uid` peut être présente dans chaque `élément` avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### `exdate`

Donne une date et une heure auxquelles l'événement ne se produit pas malgré les règles de récurrence.

La `valeur` doit être du texte qui est soit une [chaîne de date valide](#), soit une [chaîne de date et d'heure globale valide](#).

N'importe quel nombre de propriétés avec le nom `exdate` peut être présent dans chaque `élément` avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### `rdate`

Donne une date et une heure auxquelles l'événement se reproduit.

La `valeur` doit être un texte qui est l'un des suivants :

- Une [chaîne de date valide](#).
- Une [chaîne de date et d'heure globale valide](#).
- Une [chaîne de date et d'heure globale valide](#) suivie d'un caractère SOLIDUS U+002F (/) suivi d'une deuxième [chaîne de date et d'heure globale valide](#) représentant une heure ultérieure.
- Une [chaîne de date et d'heure globale valide](#) suivie d'un caractère U+002F SOLIDUS (/) suivi d'une [chaîne de durée d'événement valide](#).

N'importe quel nombre de propriétés avec le nom `rdate` peut être présent dans chaque `élément` avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### `rrule`

Donne une règle pour trouver les dates et les heures auxquelles l'événement se produit.



La [valeur](#) doit être un texte correspondant au type de valeur RECUR défini dans *iCalendar* . [\[RFC5545\]](#)

Une seule propriété avec le nom [rrule](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **created**

Indique la date et l'heure auxquelles les informations sur l'événement ont été créées pour la première fois dans un système de calendrier.

La [valeur](#) doit être un texte correspondant à une [chaîne de date et d'heure globale valide](#) .

Une seule propriété avec le nom [created](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **last-modified**

Donne la date et l'heure auxquelles les informations sur l'événement ont été modifiées pour la dernière fois dans un système de calendrier.

La [valeur](#) doit être un texte correspondant à une [chaîne de date et d'heure globale valide](#) .

Une seule propriété avec le nom [last-modified](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

#### **sequence**

Donne un numéro de révision pour les informations sur l'événement.

La [valeur](#) doit être un texte qui est un [entier non négatif valide](#) .

Une seule propriété avec le nom [sequence](#) peut être présente dans chaque [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>.

Une chaîne est une **chaîne de durée d'événement valide** si elle correspond au modèle suivant :

1. A U+0050 LETTRE MAJUSCULE LATINE P caractère (P).
2. L'un des éléments suivants :
  - Un [entier non négatif valide](#) suivi du caractère U+0057 LETTRE MAJUSCULE LATINE W (W). L'entier représente une durée de ce nombre de semaines.

- Au moins un, et éventuellement les deux dans cet ordre, parmi les suivants :
  1. Un [entier non négatif valide](#) suivi d'un caractère U+0044 LETTRE MAJUSCULE LATINE D (D). L'entier représente une durée de ce nombre de jours.
  2. A U+0054 LETTRE MAJUSCULE LATINE T caractère (T) suivi de l'un des éléments suivants, ou du premier et du deuxième des éléments suivants dans cet ordre, ou du deuxième et du troisième des éléments suivants dans cet ordre, ou des trois éléments suivants dans cet ordre:
    1. Un [entier non négatif valide](#) suivi d'un caractère U+0048 LETTRE MAJUSCULE LATINE H (H). L'entier représente une durée de ce nombre d'heures.
    2. Un [entier non négatif valide](#) suivi d'un caractère U + 004D LETTRE MAJUSCULE LATINE M (M). L'entier représente une durée de ce nombre de minutes.
    3. Un [entier non négatif valide](#) suivi d'un caractère U+0053 LETTRE MAJUSCULE LATINE S (S). L'entier représente une durée de ce nombre de secondes.

### 5.3.2.1 Conversion vers iCalendar

Étant donné une liste de nœuds *nœuds* dans un [Document](#), un agent utilisateur doit exécuter l'algorithme suivant pour **extraire toutes les données vEvent représentées par ces nœuds** :

1. Si aucun des nœuds dans *nodes* n'est un [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>, alors il n'y a pas de données vEvent. Abandonne l'algorithme, ne renvoyant rien.
2. Soit *la sortie* une chaîne vide.
3. [Ajoutez une ligne iCalendar](#) avec le type " BEGIN" et la valeur " VCALENDAR" à *la sortie* .
4. [Ajoutez une ligne iCalendar](#) avec le type " PRODID" et la valeur égale à une chaîne spécifique à l'agent utilisateur représentant l'agent utilisateur à *afficher* .
5. [Ajoutez une ligne iCalendar](#) avec le type " VERSION" et la valeur " 2.0" à *la sortie* .
6. Pour chaque nœud *node* in *nodes* qui est un [élément](#) avec le type <http://microformats.org/profile/hcalendar#vevent>, exécutez les étapes suivantes :
  1. [Ajoutez une ligne iCalendar](#) avec le type " BEGIN" et la valeur " VEVENT" à *la sortie* .

2. [Ajoutez une ligne iCalendar](#) avec le type " DTSTAMP" et une valeur consistant en une chaîne iCalendar DATE-TIME représentant la date et l'heure actuelles, avec l'annotation " VALUE=DATE-TIME", à la sortie . [\[RFC5545\]](#)
3. Pour chaque élément *element* qui est [une propriété du nœud](#) d'élément : pour chaque nom *name* dans [les noms de propriété](#) de l'élément , exécutez l'ensemble approprié de sous-étapes dans la liste suivante :

Si la [valeur](#) de la propriété est un [élément](#)

Passer la propriété.

Si la propriété est [dtend](#)

Si la propriété est [dtstart](#)

Si la propriété est [exdate](#)

Si la propriété est [rdate](#)

Si la propriété est [created](#)

Si la propriété est [last-modified](#)

Soit *value* le résultat de la suppression de tous les caractères U+002D HYPHEN-MINUS (-) et U+003A COLON (:) de la propriété [value](#) .

Si la [valeur](#) de la propriété est une [chaîne de date valide](#) , [ajoutez une ligne iCalendar](#) avec le *nom* du type et la valeur *value* à *sortir* , avec l'annotation " VALUE=DATE".

[Sinon](#), si la [valeur](#) de la propriété est une [chaîne de date et d'heure globale valide](#) , ajoutez [une ligne iCalendar](#) avec le *nom* du type et la valeur *value* à *sortir* , avec l'annotation " VALUE=DATE-TIME".

Sinon, sautez la propriété.

**Sinon**

[Ajoutez une ligne iCalendar](#) avec le *nom* du type et la [valeur](#) de la propriété à *output* .

4. [Ajoutez une ligne iCalendar](#) avec le type " END" et la valeur " VEVENT" à la sortie .
7. [Ajoutez une ligne iCalendar](#) avec le type " END" et la valeur " VCALENDAR" à la sortie .

Lorsque l'algorithme ci-dessus indique que l'agent utilisateur doit **ajouter une ligne iCalendar** composée d'un type *type* , d'une valeur *value* et éventuellement d'une annotation à une *sortie* de chaîne , il doit exécuter les étapes suivantes :

1. Soit *ligne* une chaîne vide.

2. Ajouter le type , [converti en majuscule ASCII](#) , à la ligne .
3. S'il y a une annotation :
  1. Ajoutez un caractère U+003B SEMICOLON (;) à la ligne .
  2. Ajoutez l'annotation à la ligne .
4. Ajoutez un caractère U+003A COLON (:) à la ligne .
5. Préfixez chaque caractère U+005C REVERSE SOLIDUS (\) dans la valeur avec un autre caractère U+005C REVERSE SOLIDUS (\).
6. Préfixez chaque caractère virgule U+002C (,) dans la valeur avec un caractère U+005C REVERSE SOLIDUS (\).
7. Préfixez chaque caractère U+003B SEMICOLON (;) dans la valeur avec un caractère U+005C REVERSE SOLIDUS (\).
8. Remplacez chaque paire de caractères U+000D CARRIAGE RETURN U+000A LINE FEED (CRLF) dans la valeur par un caractère U+005C REVERSE SOLIDUS (\) suivi d'un caractère U+006E LATIN SMALL LETTER N (n).
9. Remplacez chaque caractère U+000D CARRIAGE RETURN (CR) ou U+000A LINE FEED (LF) restant dans la valeur par un caractère U+005C REVERSE SOLIDUS (\) suivi d'un caractère U+006E LATIN SMALL LETTER N (n).
10. Ajouter la valeur à la ligne .
11. Laissez la longueur maximale être 75.
12. Tant que [la longueur du point de code](#) de la ligne est supérieure à la longueur maximale :
  1. Ajoutez les premiers points de code de longueur maximale de la ligne à la sortie .
  2. Supprimez les premiers points de code de longueur maximale de la ligne .
  3. Ajoutez un caractère U+000D RETOUR CHARIOT (CR) à la sortie .
  4. Ajoutez un caractère LINE FEED U+000A (LF) à la sortie .
  5. Ajoutez un caractère U+0020 SPACE à la sortie .
  6. Laissez la longueur maximale être 74.
13. Ajouter (ce qui reste de) la ligne à la sortie .
14. Ajoutez un caractère U+000D RETOUR CHARIOT (CR) à la sortie .

15. Ajoutez un caractère LINE FEED U+000A (LF) à la sortie .

Cet algorithme peut générer une sortie iCalendar non valide si l'entrée n'est pas conforme aux règles décrites pour le <http://microformats.org/profile/hcalendar#vevent> type d'élément et les noms de propriété définis .

### 5.3.2.2 Exemples

*Cette section est non normative.*

Voici un exemple de page qui utilise le vocabulaire vEvent pour baliser un événement :

```
<body itemscope
itemtype="http://microformats.org/profile/hcalendar#vevent">
...
<h1 itemprop="summary">Bluesday Tuesday: Money Road</h1>
...
<time itemprop="dtstart" datetime="2009-05-05T19:00:00Z">May 5th @
7pm</time>
(until <time itemprop="dtend" datetime="2009-05-
05T21:00:00Z">9pm</time>)
...
<a href="http://livebrum.co.uk/2009/05/05/bluesday-tuesday-money-
road"
rel="bookmark" itemprop="url">Link to this page</a>
...
<p>Location: <span itemprop="location">The RoadHouse</span></p>
...
<p><input type="button" value="Add to Calendar"
onclick="location = getCalendar(this)"></p>
...
<meta itemprop="description" content="via livebrum.co.uk">
</body>
```

La `getCalendar()` fonction est laissée en exercice au lecteur.

La même page peut proposer un balisage, comme celui-ci, pour le copier-coller dans les blogs :

```
<div itemscope
itemtype="http://microformats.org/profile/hcalendar#vevent">
<p>I'm going to
```

```

<strong itemprop="summary">Bluesday Tuesday: Money Road</strong>,
<time itemprop="dtstart" datetime="2009-05-05T19:00:00Z">May 5th
at 7pm</time>
to <time itemprop="dtend" datetime="2009-05-
05T21:00:00Z">9pm</time>,
at <span itemprop="location">The RoadHouse</span>!</p>
<p><a href="http://livebrum.co.uk/2009/05/05/bluesday-tuesday-
money-road"
itemprop="url">See this event on livebrum.co.uk</a>.</p>
<meta itemprop="description" content="via livebrum.co.uk">
</div>

```

### 5.3.3 Autorisation des travaux

Un élément avec le [type d'élément](#) <http://n.whatwg.org/work> représente une œuvre (par exemple un article, une image, une vidéo, une chanson, etc.). Ce type est principalement destiné à permettre aux auteurs d'inclure des informations de licence pour les œuvres.

Voici les [noms de propriété définis](#) du type .

#### work

Identifie le travail décrit.

La [valeur](#) doit être une [URL absolue](#) .

Exactement une propriété avec le nom [work](#) doit être présente dans chaque [élément](#) avec le type <http://n.whatwg.org/work>.

#### title

Donne le nom de l'oeuvre.

Une seule propriété avec le nom [title](#) peut être présente dans chaque [élément](#) avec le type <http://n.whatwg.org/work>.

#### author

Donne le nom ou les coordonnées d'un des auteurs ou créateurs de l'œuvre.

La [valeur](#) doit être soit un [élément](#) avec le type <http://microformats.org/profile/hcard>, soit du texte.

N'importe quel nombre de propriétés avec le nom [author](#) peut être présent dans chaque [élément](#) avec le type <http://n.whatwg.org/work>.

#### license

Identifie l'une des licences sous lesquelles l'œuvre est disponible.

La [valeur](#) doit être une [URL absolue](#) .

N'importe quel nombre de propriétés avec le nom [license](#) peut être présent dans chaque [élément](#) avec le type <http://n.whatwg.org/work>.

### 5.3.3.1 Exemples

*Cette section est non normative.*

Cet exemple montre une image intégrée intitulée *My Pond* , sous licence internationale Creative Commons Attribution-Share Alike 4.0 et sous licence MIT simultanément.

```
<figure itemscope itemtype="http://n.whatwg.org/work">
  
  <figcaption>
    <p><cite itemprop="title">My Pond</cite></p>
    <p><small>Licensed under the <a itemprop="license"
      href="https://creativecommons.org/licenses/by-sa/4.0/">Creative
      Commons Attribution-Share Alike 4.0 International License</a>
      and the <a itemprop="license"
      href="http://www.opensource.org/licenses/mit-license.php">MIT
      license</a>.</small>
    </figcaption>
  </figure>
```

## 5.4 Conversion de HTML vers d'autres formats

### 5.4.1 JSON

Étant donné une liste de nœuds *nœuds* dans un [Document](#), un agent utilisateur doit exécuter l'algorithme suivant pour **extraire les microdonnées de ces nœuds dans un formulaire JSON** :

1. Soit *result* un objet vide.
2. Soit *items* un tableau vide.

3. Pour chaque *nœud* dans *nodes* , vérifiez si l'élément est un [élément de microdonnées de niveau supérieur](#) , et si c'est le cas, [récupérez l'objet](#) pour cet élément et ajoutez-le aux *items* .
4. Ajoutez une entrée au *résultat* appelée " *items* " dont la valeur est les *éléments* du tableau .
5. Renvoyez le résultat de la sérialisation du *résultat* à JSON de la manière la plus courte possible (c'est-à-dire sans espace entre les jetons, pas de chiffres zéro inutiles dans les nombres et en utilisant uniquement les échappements Unicode dans les chaînes pour les caractères qui n'ont pas de séquence d'échappement dédiée), et avec une minuscule " e " utilisé, le cas échéant, dans la représentation de n'importe quel nombre. [\[JSON\]](#)

*Cet algorithme renvoie un objet avec une seule propriété qui est un tableau, au lieu de simplement renvoyer un tableau, de sorte qu'il est possible d'étendre l'algorithme à l'avenir si nécessaire.*

Lorsque l'agent utilisateur doit **obtenir l'objet** pour un élément *item* , éventuellement avec une liste d'éléments *memory* , il doit exécuter les sous-étapes suivantes :

1. Soit *result* un objet vide.
2. Si aucune *mémoire* n'a été transmise à l'algorithme, laissez *la mémoire* être une liste vide.
3. Ajouter un *élément* à *la mémoire* .
4. Si l' *élément* a des [types d'éléments](#) , ajoutez une entrée au *résultat* appelée " *type* " dont la valeur est un tableau répertoriant les [types d'éléments](#) d' *élément* , dans l'ordre dans lequel ils ont été spécifiés sur l' [itemtype](#) attribut.
5. Si l' *élément* a un [identifiant global](#) , ajoutez une entrée au *résultat* appelée " *id* " dont la valeur est l' [identifiant global](#) de l'*élément* .
6. Soit *properties* un objet vide.
7. Pour chaque élément *element* qui a un ou plusieurs [noms de propriété](#) et qui est l'une [des propriétés de l'item](#) *item* , dans l'ordre dans lequel ces éléments sont donnés par l'algorithme qui renvoie [les propriétés d'un item](#) , exécutez les sous-étapes suivantes :
  1. Soit *value* la [valeur de propriété](#) de *element* .
  2. Si *value* est un [élément](#) , alors : Si *value* est en *mémoire* , alors laissez *value* être la chaîne " ERROR ". Sinon, [récupérez l'objet](#) pour *value* , en transmettant une copie de *memory* , puis remplacez *value* par l'objet renvoyé par ces étapes.



3. Pour chaque nom *name* dans [les noms de propriété](#) de l'élément , exécutez les sous-étapes suivantes :
  1. S'il n'y a pas d'entrée nommée *name* dans *properties* , ajoutez une entrée nommée *name* aux *propriétés* dont la valeur est un tableau vide.
  2. Ajoutez *value* à l'entrée nommée *name* dans *properties* .
8. Ajoutez une entrée au *résultat* appelée " *properties* " dont la valeur est les *propriétés* de l' objet .
9. Retourner *le résultat* .

Par exemple, prenez ce balisage :

```
<!DOCTYPE HTML>
<html lang="en">
<title>My Blog</title>
<article itemscope itemtype="http://schema.org/BlogPosting">
  <header>
    <h1 itemprop="headline">Progress report</h1>
    <p><time itemprop="datePublished" datetime="2013-08-29">today</time></p>
    <link itemprop="url" href="?comments=0">
  </header>
  <p>All in all, he's doing well with his swim lessons. The biggest thing was he had trouble putting his head in, but we got it down.</p>
  <section>
    <h1>Comments</h1>
    <article itemprop="comment" itemscope itemtype="http://schema.org/UserComments" id="c1">
      <link itemprop="url" href="#c1">
      <footer>
        <p>Posted by: <span itemprop="creator" itemscope itemtype="http://schema.org/Person">
          <span itemprop="name">Greg</span>
        </span></p>
        <p><time itemprop="commentTime" datetime="2013-08-29">15 minutes ago</time></p>
      </footer>
      <p>Ha!</p>
    </article>
    <article itemprop="comment" itemscope itemtype="http://schema.org/UserComments" id="c2">
```

```

    <link itemprop="url" href="#c2">
  </footer>
  <p>Posted by: <span itemprop="creator" itemscope
itemtype="http://schema.org/Person">
    <span itemprop="name">Charlotte</span>
  </span></p>
  <p><time itemprop="commentTime" datetime="2013-08-29">5 minutes
ago</time></p>
</footer>
  <p>When you say "we got it down"...</p>
</article>
</section>
</article>

```

Il serait transformé en JSON suivant par l'algorithme ci-dessus (en supposant que l'URL de la page était <https://blog.example.com/progress-report>):

```

{
  "items": [
    {
      "type": [ "http://schema.org/BlogPosting" ],
      "properties": {
        "headline": [ "Progress report" ],
        "datePublished": [ "2013-08-29" ],
        "url": [ "https://blog.example.com/progress-
report?comments=0" ],
        "comment": [
          {
            "type": [ "http://schema.org/UserComments" ],
            "properties": {
              "url": [ "https://blog.example.com/progress-
report#c1" ],
              "creator": [
                {
                  "type": [ "http://schema.org/Person" ],
                  "properties": {
                    "name": [ "Greg" ]
                  }
                }
              ],
              "commentTime": [ "2013-08-29" ]
            }
          }
        ]
      }
    }
  ],

```

```

{
  "type": [ "http://schema.org/UserComments" ],
  "properties": {
    "url": [ "https://blog.example.com/progress-report#c2" ],
    "creator": [
      {
        "type": [ "http://schema.org/Person" ],
        "properties": {
          "name": [ "Charlotte" ]
        }
      }
    ],
    "commentTime": [ "2013-08-29" ]
  }
}

```

## 6 Interaction avec l'utilisateur

### 6.1 L' hiddenattribut



Tous [les éléments HTML](#) peuvent avoir l' **hidden**attribut content défini. L' **hidden**attribut est un [attribut énuméré](#) . Le tableau suivant répertorie les états de cet attribut :

État	Mots clés
<a href="#">Jusqu'à l'état retrouvé</a>	<b>until-found</b>
<a href="#">État caché</a>	La chaîne vide
	<b>hidden</b>

L'attribut peut être omis. La [valeur par défaut invalide](#) est l' [état caché](#) . La [valeur manquante par défaut](#) est l' **état non caché** .

Lorsqu'un élément a l' `hidden` attribut à l' **état masqué** , cela indique que l'élément n'est pas encore, ou n'est plus, directement pertinent pour l'état actuel de la page, ou qu'il est utilisé pour déclarer du contenu à réutiliser par d'autres parties du page au lieu d'être directement accessible par l'utilisateur. Les agents utilisateurs ne doivent pas rendre les éléments qui sont dans l' **état masqué** . Cette exigence peut être mise en œuvre indirectement via la couche de style. Par exemple, un navigateur Web pourrait implémenter ces exigences [en utilisant les règles suggérées dans la section Rendu](#) .

Lorsqu'un élément a l' `hidden` attribut dans l' **état caché jusqu'à trouvé** , cela indique que l'élément est caché comme l' **état caché** mais que le contenu à l'intérieur de l'élément sera accessible à [la recherche dans la page](#) et [à la navigation par fragment](#) . Lorsque ces fonctionnalités tentent de défiler jusqu'à une cible qui se trouve dans le sous-arbre de l'élément, l'agent utilisateur supprimera l' `hidden` attribut afin de révéler le contenu avant de défiler jusqu'à celui-ci. En plus de supprimer l' `hidden` attribut, un événement nommé `beforematch` est également déclenché sur l'élément avant la `hidden` suppression de l'attribut.

Les navigateurs Web utiliseront 'content-visibility: hidden' au lieu de 'display: none' lorsque l' `hidden` attribut est dans l' **état caché jusqu'à ce qu'il soit trouvé** , comme spécifié dans la [section Rendu](#) .

*Étant donné que cet attribut est généralement implémenté à l'aide de CSS, il est également possible de le remplacer à l'aide de CSS. Par exemple, une règle qui applique 'display: block' à tous les éléments annulera les effets de l' état **caché** . Les auteurs doivent donc faire attention lors de l'écriture de leurs feuilles de style pour s'assurer que l'attribut est toujours stylisé comme prévu. De plus, les anciens agents utilisateurs qui ne prennent pas en charge l' **état caché jusqu'à ce qu'ils soient trouvés** auront 'display: none' au lieu de 'content-visibility: hidden', les auteurs sont donc encouragés à s'assurer que leurs feuilles de style ne changent pas le ' display' ou 'content-visibility' des éléments **cachés jusqu'à ce qu'ils soient trouvés** .*

*Puisque les éléments avec l' `hidden` attribut dans l' **état caché jusqu'à trouvé** utilisent 'content-visibility: hidden' au lieu de 'display: none', il y a deux mises en garde concernant l' **état caché jusqu'à trouvé** qui le différencient de l' **état caché** :*

- 1. L'élément doit être affecté par [le confinement de la mise en page](#) afin d'être révélé par la recherche dans la page. Cela signifie que si l'élément caché [jusqu'à ce qu'il soit trouvé](#) a une valeur 'display' de 'none', 'contents' ou 'inline', alors l'élément ne sera pas révélé par find-in-page.*
- 2. L'élément aura toujours une [boîte générée](#) lorsqu'il est [caché jusqu'à ce qu'il soit trouvé](#) , ce qui signifie que les bordures, la marge et le remplissage seront toujours rendus autour de l'élément.*

Dans l'exemple de squelette suivant, l'attribut est utilisé pour masquer l'écran principal du jeu Web jusqu'à ce que l'utilisateur se connecte :

```
<h1>The Example Game</h1>
<section id="login">
```

```

<h2>Login</h2>
<form>
  ...
  <!-- calls login() once the user's credentials have been
checked -->
</form>
<script>
  function login() {
    // switch screens
    document.getElementById('login').hidden = true;
    document.getElementById('game').hidden = false;
  }
</script>
</section>
<section id="game" hidden>
  ...
</section>

```

L' hidden attribut ne doit pas être utilisé pour masquer un contenu qui pourrait légitimement être affiché dans une autre présentation. Par exemple, il est incorrect d'utiliser hidden pour masquer les panneaux dans une boîte de dialogue à onglets, car l'interface à onglets est simplement une sorte de présentation de débordement - on pourrait tout aussi bien afficher tous les contrôles de formulaire dans une grande page avec une barre de défilement. Il est également incorrect d'utiliser cet attribut pour masquer le contenu d'une seule présentation — si quelque chose est marqué hidden, il est masqué de toutes les présentations, y compris, par exemple, des lecteurs d'écran.

Les éléments qui ne sont pas eux-mêmes hidden ne doivent pas [créer de lien hypertexte](#) vers des éléments qui sont hidden. Les `for` attributs de `label` et `output` les éléments qui ne sont pas eux-mêmes hidden ne doivent pas non plus faire référence à des éléments qui sont hidden. Dans les deux cas, de telles références seraient source de confusion pour l'utilisateur.

Les éléments et les scripts peuvent cependant faire référence à des éléments qui se trouvent hidden dans d'autres contextes.

Par exemple, il serait incorrect d'utiliser l' href attribut pour créer un lien vers une section marquée avec l' hidden attribut. Si le contenu n'est pas applicable ou pertinent, il n'y a aucune raison de créer un lien vers celui-ci.

Ce serait bien, cependant, d'utiliser l' aria-describedby attribut ARIA pour faire référence à des descriptions qui sont elles-mêmes hidden. Bien que cacher les descriptions implique qu'elles ne sont pas utiles seules, elles pourraient être écrites

de telle manière qu'elles soient utiles dans le contexte spécifique d'être référencées à partir des éléments qu'elles décrivent.

De même, un [canvas](#) élément avec l' [hidden](#) attribut pourrait être utilisé par un moteur graphique scripté comme tampon hors écran, et un contrôle de formulaire pourrait faire référence à un [form](#) élément masqué à l'aide de son [form](#) attribut.

Les éléments d'une section masqués par l' [hidden](#) attribut sont toujours actifs, par exemple les scripts et les contrôles de formulaire dans ces sections s'exécutent et se soumettent respectivement. Seule leur présentation à l'utilisateur change.



Les [hidden](#) étapes du getter sont :

1. Si l' [hidden](#) attribut est dans l' état [non trouvé](#) , alors renvoie " [until-found](#)".
2. Si l' [hidden](#) attribut est défini, renvoie true.
3. Renvoie faux.

Les [hidden](#) étapes du setter sont :

1. Si la valeur donnée est une chaîne correspondant à une correspondance [ASCII non sensible à la casse](#) pour " [until-found](#)", définissez l' [hidden](#) attribut sur " [until-found](#)".
2. Sinon, si la valeur donnée est fausse, supprimez l' [hidden](#) attribut.
3. Sinon, si la valeur donnée est la chaîne vide, supprimez l' [hidden](#) attribut.
4. Sinon, si la valeur donnée est nulle, supprimez l' [hidden](#) attribut.
5. Sinon, si la valeur donnée est 0, supprimez l' [hidden](#) attribut.
6. Sinon, si la valeur donnée est NaN, supprimez l' [hidden](#) attribut.
7. Sinon, définissez l' [hidden](#) attribut sur la chaîne vide.

**L' algorithme de révélation de l'ancêtre caché jusqu'à ce qu'il soit trouvé** consiste à exécuter les étapes suivantes sur *currentNode* :

1. Alors que *currentNode* a un nœud parent dans l' [arborescence plate](#) :
  1. Si *currentNode* a l' [hidden](#) attribut caché [jusqu'à ce qu'il](#) soit trouvé, alors :
    1. [Lancez un événement](#) nommé [beforematch](#) à *currentNode* .

2. Supprimez l' hidden attribut de *currentNode* .
2. Définissez *currentNode* sur le nœud parent de *currentNode* dans l' arborescence plate .

## 6.2 Visibilité des pages

L' état de visibilité du système d' un élément navigable traversable , y compris sa valeur initiale lors de la création, est déterminé par l'agent utilisateur. Il indique, par exemple, si la fenêtre du navigateur est réduite, si un onglet du navigateur est actuellement en arrière-plan ou si un élément système tel qu'un sélecteur de tâches masque la page.

Lorsqu'un agent utilisateur détermine que l' état de visibilité du système pour traversable navigable *traversable* est passé à *newState* , il doit exécuter les étapes suivantes :

1. Soit *navigables* les navigables descendants inclusifs de *traversable* .
2. Pour chaque *navigable* des *navigables* **dans quel ordre ?** :
  1. Soit *document* le document actif de *navigable* .
  2. Mettre en file d'attente une tâche globale sur la source de tâche d'interaction utilisateur en fonction de l'objet global pertinent du document pour mettre à jour l'état de visibilité du *document* avec *newState* .

A Document a un **état de visibilité** , qui est " *hidden* " ou " *visible* ", initialement défini sur " *hidden* ".



Les **visibilityState** étapes du getter consistent à retourner cet état de visibilité .



Les **hidden** étapes du getter consistent à retourner true si l'état de visibilité de cet est " *hidden* ", sinon false.hidden

Pour **mettre à jour l'état de visibilité** du Document *document* à *visibilityState* :

1. Si l'état de visibilité du *document* est égal à *visibilityState* , alors retournez.
2. Définissez l'état de visibilité du *document* sur *visibilityState* .

3. Exécutez les [étapes de changement d'orientation de l'écran](#) avec *document* . [\[ORIENTATION DE L'ÉCRAN\]](#)
4. Exécutez toutes **les étapes de changement de visibilité de page** qui peuvent être définies dans d'autres spécifications, avec [état de visibilité](#) et *document* .

Il serait préférable que les auteurs de spécifications envoient une demande d'extraction pour ajouter directement des appels à partir d'ici dans leurs spécifications, au lieu d'utiliser le hook [d'étapes de modification de la visibilité de la page](#) , afin de garantir un ordre d'appel de spécification croisée bien défini. Au moment de la rédaction de cet article, les spécifications suivantes sont connues pour avoir [des étapes de modification de la visibilité des pages](#) , qui seront exécutées dans un ordre non spécifié : *API Device Posture* et *Web NFC* . [\[POSTURE DE L'APPAREIL\]](#) [\[WEBNFC\]](#)

5. Lancez un événement nommé [visibilitychange](#) à *document* , avec son [bubbles](#) attribut initialisé à true.

## 6.3 Sous-arbres inertes

Voir aussi [inert](#) pour une explication de l'attribut du même nom.

Un nœud (en particulier des éléments et des nœuds de texte) peut être **inerte** . Lorsqu'un nœud est [inerte](#) :

- Le test d'atteinte doit agir comme si la propriété CSS ['pointer-events'](#) était [définie sur 'none'](#).
- La fonctionnalité de sélection de texte doit agir comme si la propriété CSS ['user-select'](#) était définie sur 'none'.
- S'il est [modifiable](#) , le nœud se comporte comme s'il n'était pas modifiable.
- L'agent utilisateur peut ignorer le nœud pour les besoins de [find-in-page](#) .

*Les nœuds inertes ne peuvent généralement pas être focalisés. Les nœuds inertes qui sont [des commandes](#) seront également désactivés.*

Cependant, les agents utilisateurs peuvent permettre à l'utilisateur d'outrepasser les restrictions sur la recherche et la sélection de texte.

Par défaut, un nœud n'est pas [inerte](#) .



### 6.3.1 Dialogues modaux et sous-arbres inertes

Un Document *document* est **bloqué par un** *sujet* de dialogue modal si *le sujet* est l' dialogélément le plus haut dans la couche supérieure du *document* . Tant que *document* est ainsi bloqué, chaque nœud connecté à *document* , à l'exception de l' élément *sujet* et de ses descendants d'arborescence plate , doit devenir inerte .

*le sujet* peut en outre devenir inerte via l' inertattribut, mais seulement s'il est spécifié sur *le sujet* lui-même (c'est-à-dire que *le sujet* échappe à l'inertie des ancêtres) ; Les descendants de l'arbre plat du *sujet* peuvent devenir inertes de la même manière.

*La méthode dialogde l'élément showModal() provoque le déclenchement de ce mécanisme, en ajoutant l' dialogélément à la couche supérieure de son nœud document .*

### 6.3.2 L' inertattribut

L' inertattribut est un attribut booléen qui indique, par sa présence, que l'élément et tous ses descendants d'arbres plats qui n'échappent pas autrement à l'inertie (comme les dialogues modaux) doivent être rendus inertes par l'agent utilisateur.

*Par défaut, il n'y a aucune indication visuelle persistante d'un sous-arbre inerte. Les auteurs sont encouragés à indiquer clairement quelles parties de leur document sont actives et lesquelles sont inertes, afin d'éviter toute confusion chez l'utilisateur. En particulier, il convient de rappeler que tous les utilisateurs ne peuvent pas voir toutes les parties d'une page à la fois ; par exemple, les utilisateurs de lecteurs d'écran, les utilisateurs de petits appareils ou de loupes, et même les utilisateurs utilisant des fenêtres particulièrement petites pourraient ne pas être en mesure de voir la partie active d'une page et pourraient être frustrés si les sections inertes ne sont pas manifestement inertes. Pour les contrôles individuels, l' disabledattribut est probablement plus approprié.*



L' inertattribut IDL doit refléter l'attribut de contenu du même nom.

## 6.4 Suivi de l'activation de l'utilisateur

Pour éviter l'utilisation abusive de certaines API qui pourraient gêner les utilisateurs (par exemple, ouvrir des fenêtres contextuelles ou faire vibrer les téléphones), les agents utilisateurs n'autorisent ces API que lorsque l'utilisateur interagit activement avec la page Web ou a interagi avec la page au moins une fois. Cet état "d'interaction active" est maintenu par les mécanismes définis dans cette section.

### 6.4.1 Modèle de données

Afin de suivre l'activation de l'utilisateur, chaque Window  $W$  a un **horodatage de dernière activation**. Il s'agit d'un nombre indiquant la dernière fois que  $W$  a reçu une notification d'activation. Il correspond à une DOMHighResTimeStamp valeur sauf dans deux cas : l'infini positif indique que  $W$  n'a jamais été activé, tandis que l'infini négatif indique qu'une API activée par l'utilisateur a consommé la dernière activation utilisateur de  $W$ . La valeur initiale est l'infini positif.

Un agent utilisateur définit également une **durée d'activation transitoire**, qui est un nombre constant indiquant la durée pendant laquelle une activation utilisateur est disponible pour certaines API d'activation utilisateur (par exemple, pour l'ouverture de fenêtres contextuelles).

*La durée d'activation transitoire est prévue d'au plus quelques secondes, afin que l'utilisateur puisse éventuellement percevoir le lien entre une interaction avec la page et la page appelant l'API activation-gated.*

Ces deux valeurs impliquent deux états booléens d'activation de l'utilisateur pour  $W$  :

#### **Activation collante**

Lorsque le temps de haute résolution actuel donné à  $W$  est supérieur ou égal au dernier horodatage d'activation dans  $W$ , on dit que  $W$  a une activation persistante.

Il s'agit de l'état d'activation historique de  $W$ , indiquant si l'utilisateur a déjà interagi dans  $W$ . Il commence par false, puis devient true (et ne redevient jamais false) lorsque  $W$  reçoit la toute première notification d'activation.

#### **Activation transitoire**

Lorsque le temps de haute résolution actuel donné  $W$  est supérieur ou égal au dernier horodatage d'activation dans  $W$ , et inférieur au dernier horodatage d'activation dans  $W$  plus la durée d'activation transitoire, on dit alors que  $W$  a une activation transitoire.

Il s'agit de l'état d'activation actuel de  $W$ , indiquant si l'utilisateur a récemment interagi dans  $W$ . Cela commence par une valeur fausse et reste vrai pendant un temps limité après chaque notification d'activation que  $W$  reçoit.

L'état d'activation transitoire est considéré comme expiré s'il devient faux parce que la durée d'activation transitoire s'est écoulée depuis la dernière activation de l'utilisateur. A noter qu'il peut devenir faux même avant l'heure d'expiration via une consommation d'activation.

*L' horodatage de la dernière activation est conservé même après que le Document change son statut entièrement actif (par exemple, après avoir quitté un Document, ou avoir navigué vers un mis en cache Document). Cela signifie que l'état d'activation persistant s'étend sur plusieurs navigations tant que le même Document est réutilisé. Pour l'état d'activation transitoire, le*

délai d'expiration d'origine reste inchangé (c'est-à-dire que l'état expire toujours dans la limite de durée d'activation transitoire à partir de l' événement d'entrée déclenchant l'activation d'origine ). Il est important d'en tenir compte lorsque vous décidez de baser certaines choses sur une activation collante ou activation transitoire .

## 6.4.2 Modèle de traitement

Lorsqu'une interaction de l'utilisateur dans un provoque le déclenchement d'un événement d'entrée déclenchant l'activation dans un Document document , l'agent utilisateur doit effectuer les étapes **de notification d'activation suivantes** avant de distribuer l'événement :

1. Assert : le document est entièrement actif .
2. Soit windows « l'objet global pertinent du document ».
3. Étendre les fenêtres avec la fenêtre active de chacun des navigables ancêtres du document .
4. Étendre les fenêtres avec la fenêtre active de chacun des éléments navigables descendants du document , filtrés pour inclure uniquement les éléments navigables dont l'origine du document actif est la même origine que l'origine du document .
5. Pour chaque fenêtre dans windows , définissez l' horodatage de la dernière activation de la fenêtre sur l' heure haute résolution actuelle .

Un **événement d'entrée déclencheur d'activation** est tout événement dont isTrusted l'attribut est vrai et dont type l'un des éléments suivants :

- keydown, à condition que la clé ne soit ni la Esc clé ni une touche de raccourci réservées par l'agent utilisateur.
- mousedown.
- pointerdown, à condition que l'événement pointerType soit "mouse".
- pointerup, à condition que l'événement pointerType ne soit pas "mouse".
- touchend.

Les API consommatrices d'activation définies dans cette spécification et d'autres peuvent **consommer l'activation de l'utilisateur** en effectuant les étapes suivantes, étant donné un Window *W* :

1. Si le navigable de *W* est nul, alors retournez.
2. Soit *top* le traversable de niveau supérieur du navigable de *W* .
3. Soit *navigables* les navigables descendants inclusifs du document actif de *top* .

4. Soit `windows` la liste des `Window` objets construits en prenant la [fenêtre active](#) de chaque élément dans `navigables`.
5. [Pour chaque](#) fenêtre dans `windows`, si [l'horodatage](#) de la dernière activation de la fenêtre n'est pas l'infini positif, alors définissez l'horodatage de [la dernière activation de](#) la fenêtre sur l'infini négatif.

*Notez l'asymétrie dans les ensembles de [contextes de navigation](#) de la page concernés par une [notification d'activation](#) vs une [consommation d'activation](#) : une consommation d'activation change (à faux) les états [d'activation transitoires](#) pour tous les contextes de navigation de la page, mais une notification d'activation change (à true) les états d'un sous-ensemble de ces contextes de navigation. Le caractère exhaustif de la consommation est ici délibéré : il empêche les sites malveillants d'effectuer plusieurs appels à une [API consommatrice d'activation](#) à partir d'une seule activation utilisateur (éventuellement en exploitant une hiérarchie profonde de [iframes](#)).*

### 6.4.3 API fermées par l'activation de l'utilisateur

Les API qui dépendent de l'activation de l'utilisateur sont classées en trois niveaux différents. Les niveaux sont les suivants, triés selon leur "force de dépendance" à l'activation de l'utilisateur (du plus faible au plus fort) :

#### **API d'activation bloquées**

Ces API nécessitent que l'état [d'activation persistant](#) soit vrai, elles sont donc bloquées jusqu'à la toute première activation de l'utilisateur.

#### **API d'activation transitoire**

Ces API exigent que l'état [d'activation transitoire](#) soit vrai, mais elles ne le [consomment](#) pas, de sorte que plusieurs appels sont autorisés par activation de l'utilisateur jusqu'à ce que l'état transitoire [expire](#).

#### **API consommatrices d'activations transitoires**

Ces API nécessitent que l'état [d'activation transitoire](#) soit vrai et [consomment l'activation de l'utilisateur](#) dans chaque appel pour empêcher l'activation de plusieurs appels par utilisateur.

### 6.4.4 L' [UserActivation](#) interface



Chacun `Window` a un associé `UserActivation`, qui est un [UserActivation](#) objet. Lors de la création de l' `Window` objet,

son [associé](#) `UserActivation` doit être défini sur un [nouvel](#) `UserActivation` objet créé dans le [domaine pertinent](#) `Window` de l'objet .

```
[Exposed=Window]

interface UserActivation {

    readonly attribute boolean hasBeenActive;

    readonly attribute boolean isActive;

};

partial interface Navigator {

    [SameObject] readonly attribute UserActivation userActivation;

};
```

#### **navigator.userActivation.hasBeenActive**

Retourne si la fenêtre a [une activation persistante](#) .

#### **navigator.userActivation.isActive**

Retourne si la fenêtre a [une activation transitoire](#) .



Les étapes [du](#) `userActivation` getter consistent à renvoyer [ceci](#) associé `.UserActivation`



Les `hasBeenActive` étapes du getter consistent à renvoyer true si [cet](#) objet [global pertinent](#) a [une activation persistante](#) , et false sinon.



Les `isActive` étapes du getter consistent à renvoyer true si [cet](#) objet [global pertinent](#) a [une activation transitoire](#) , et false sinon.

## 6.5 Comportement d'activation des éléments

Certains éléments en HTML ont un [comportement d'activation](#) , ce qui signifie que l'utilisateur peut les activer. Cela est toujours causé par un [click](#) événement.

L'agent utilisateur doit permettre à l'utilisateur de déclencher manuellement des éléments qui ont un [comportement d'activation](#) , par exemple en utilisant le clavier ou la voix, ou par des clics de souris. Lorsque l'utilisateur déclenche un élément avec un [comportement d'activation](#) défini autrement qu'en cliquant dessus, l'action par défaut de l'événement d'interaction doit être de [déclencher un click](#) événement sur l'élément.

```
element.click()
```

✓

Agit comme si l'élément avait été cliqué.

Chaque élément est associé à un **indicateur de clic en cours** , qui est initialement désactivé.

La [click\(\)](#) méthode doit exécuter les étapes suivantes :

1. Si cet élément est un contrôle de formulaire qui est [désactivé](#) , alors retournez.
2. Si l'[indicateur de clic en cours](#) de cet élément est défini, alors retour.
3. Définissez l'[indicateur de clic en cours](#) de cet élément .
4. [Déclenchez un événement de pointeur synthétique](#) nommé [click](#) sur cet élément, avec l' *indicateur non approuvé* défini.
5. [Désactivez l'indicateur de clic en cours](#) de cet élément .

## 6.6 Mise au point

### 6.6.1 Présentation

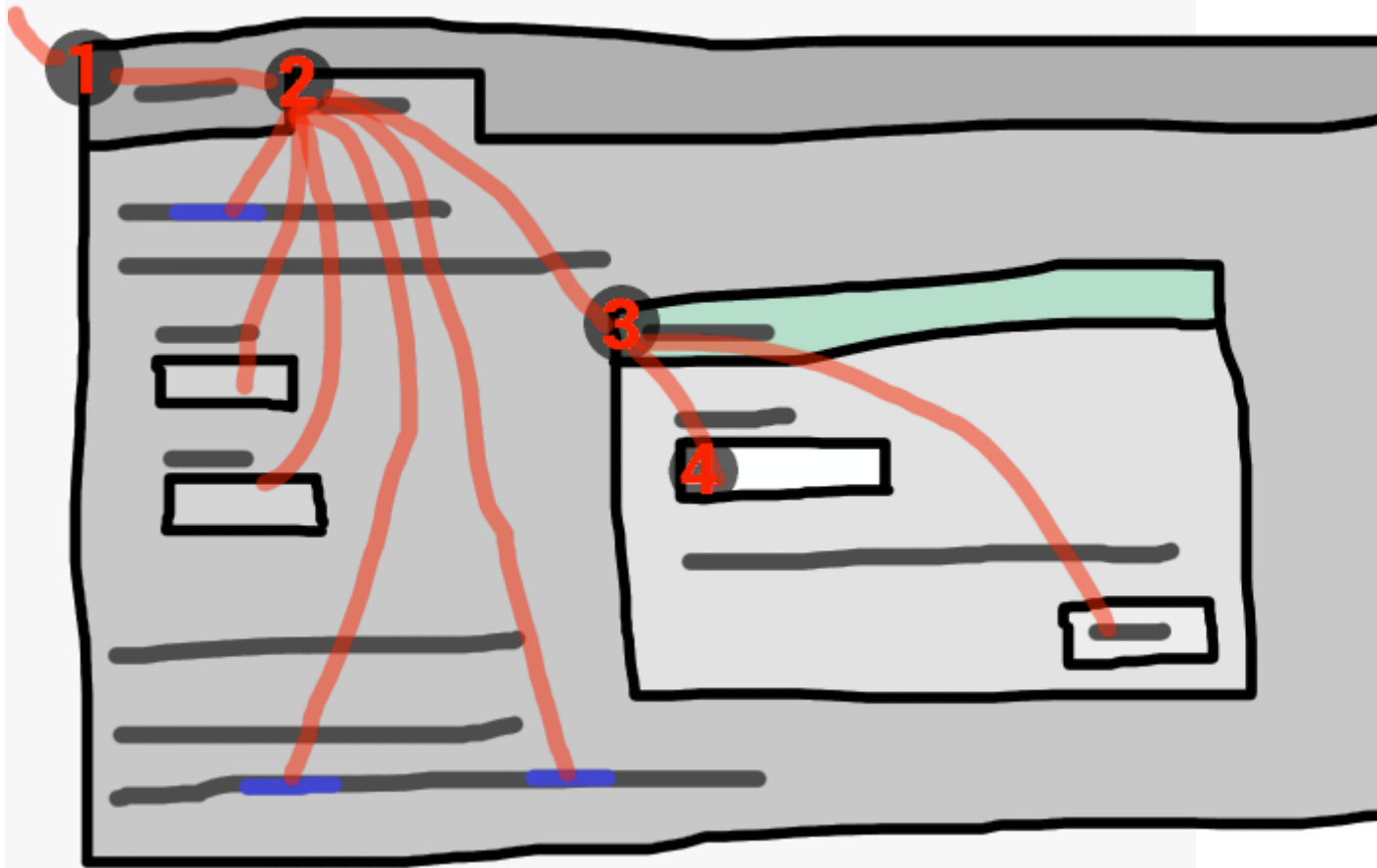
*Cette section est non normative.*

Une interface utilisateur HTML se compose généralement de plusieurs widgets interactifs, tels que des contrôles de formulaire, des zones de défilement, des liens, des boîtes de dialogue, des onglets de navigateur, etc. Ces widgets forment une hiérarchie, certains (par exemple, les onglets du navigateur, les boîtes de dialogue) en contenant d'autres (par exemple, les liens, les contrôles de formulaire).

Lors de l'interaction avec une interface à l'aide d'un clavier, la saisie des touches est acheminée du système, à travers la hiérarchie des widgets interactifs, vers un widget actif, qui est dit [focalisé](#) .

Considérez une application HTML s'exécutant dans un onglet de navigateur s'exécutant dans un environnement graphique. Supposons que cette application ait une page avec des contrôles de texte et des liens, et affiche actuellement une boîte de dialogue modale, qui a elle-même un contrôle de texte et un bouton.

La hiérarchie des widgets focalisables, dans ce scénario, inclurait la fenêtre du navigateur, qui aurait, parmi ses enfants, l'onglet du navigateur contenant l'application HTML. L'onglet lui-même aurait pour enfants les divers liens et contrôles de texte, ainsi que la boîte de dialogue. La boîte de dialogue elle-même aurait pour enfants le contrôle de texte et le bouton.



Si le widget avec le [focus](#) dans cet exemple était le contrôle de texte dans la boîte de dialogue, alors l'entrée clé serait canalisée du système graphique vers ① le navigateur Web, puis vers ② l'onglet, puis vers ③ la boîte de dialogue, et enfin vers ④ le contrôle du texte.

Les événements de clavier ciblent toujours cet élément [ciblé](#).

## 6.6.2 Modèle de données



Un [traversable de niveau supérieur](#) a le **focus système** lorsqu'il peut recevoir une entrée au clavier canalisée depuis le système d'exploitation, éventuellement destinée à l'un des [navigables descendants](#) de son [document actif](#) .

*Le focus système est perdu lorsqu'une fenêtre de navigateur perd le focus, mais peut également être perdu pour d'autres widgets système dans la fenêtre du navigateur, comme une barre d'URL.*

Le terme **zone focalisable** est utilisé pour désigner les régions de l'interface qui peuvent en outre devenir la cible d'une telle entrée au clavier. Les zones focalisables peuvent être des éléments, des parties d'éléments ou d'autres régions gérées par l'agent utilisateur.

Chaque [zone focalisable](#) a une **ancre DOM** , qui est un [Node](#) objet qui représente la position de la [zone focalisable](#) dans le DOM. (Lorsque la [zone focalisable](#) est elle-même un [Node](#), il s'agit de sa propre [ancre DOM](#) .) L' [ancre DOM](#) est utilisée dans certaines API comme substitut de la [zone focalisable](#) lorsqu'il n'y a pas d'autre objet DOM pour représenter la [zone focalisable](#) .

Le tableau suivant décrit les objets pouvant être [des zones focalisables](#) . Les cellules de la colonne de gauche décrivent des objets qui peuvent être [des zones focalisables](#) ; les cellules de la colonne de droite décrivent les [ancres DOM](#) pour ces éléments. (Les cellules qui couvrent les deux colonnes sont des exemples non normatifs.)

<a href="#">Zone focalisable</a>	<a href="#">Ancre DOM</a>
Exemples	
<p>Éléments répondant à tous les critères suivants :</p> <ul style="list-style-type: none"> <li>la <a href="#">valeur tabindex</a> de l'élément n'est pas nulle, ou l'élément est déterminé par l'agent utilisateur comme étant focalisable ;</li> <li>soit l'élément n'est pas un <a href="#">hôte fantôme</a> , soit il a une <a href="#">racine fantôme</a> dont <a href="#">le focus des délégués</a> est faux ;</li> <li>l'élément n'est pas <a href="#">réellement désactivé</a> ;</li> <li>l'élément n'est pas <a href="#">inerte</a> ;</li> <li>l'élément est <a href="#">en cours de rendu</a> ou <a href="#">utilisé en tant que contenu de secours de canevas pertinent</a> .</li> </ul>	L'élément lui-même.
<a href="#">iframe</a> , <a href="#">dialog</a> , <a href="#">&lt;input type=text&gt;</a> , parfois <a href="#">&lt;a href=""&gt;</a> (selon les conventions de la plate-forme).	
Formes des <a href="#">area</a> éléments d'une <a href="#">image cliquable</a> associée à un <a href="#">img</a> élément en cours <a href="#">de rendu</a> et qui n'est pas <a href="#">inerte</a> .	L' <a href="#">img</a> élément.



<a href="#">Zone focalisable</a>	<a href="#">Ancre DOM</a>
<b>Exemples</b>	
<p>Dans l'exemple suivant, l' <a href="#">area</a>élément crée deux formes, une sur chaque image. L' <a href="#">ancre DOM</a> de la première forme est le premier <a href="#">img</a>élément et l' <a href="#">ancre DOM</a> de la deuxième forme est le deuxième <a href="#">img</a>élément.</p> <pre> &lt;map id=wallmap&gt;&lt;area alt="Enter Door" coords="10,10,100,200" href="door.html"&gt;&lt;/map&gt; ... &lt;img src="images/innerwall.jpeg" alt="There is a white wall here, with a door." usemap="#wallmap"&gt; ... &lt;img src="images/outerwall.jpeg" alt="There is a red wall here, with a door." usemap="#wallmap"&gt; </pre>	
<b>L'agent utilisateur a fourni des sous-widgets d'éléments qui sont <a href="#">rendus</a> et qui ne sont pas <a href="#">réellement désactivés</a> ou <a href="#">inertes</a> .</b>	L'élément pour lequel la <a href="#">zone focalisable</a> est un sous-widget.
Les <a href="#">commandes de l'interface utilisateur</a> d'un <a href="#">video</a> élément, les boutons haut et bas dans une version à commande par rotation de <code>&lt;input type=number&gt;</code> , la partie du <a href="#">details</a> rendu d'un élément qui permet d'ouvrir ou de fermer l'élément à l'aide d'une saisie au clavier.	
<b>Les régions défilantes des éléments qui sont <a href="#">rendus</a> et qui ne sont pas <a href="#">inertes</a> .</b>	L'élément pour lequel la boîte que la région défilante fait défiler a été créée.
La valeur 'scroll' de la propriété CSS ' <a href="#">overflow</a> ' crée généralement une région défilable.	
<b>La <a href="#">fenêtre d'affichage</a> de a qui a un <a href="#">contexte de navigation</a><a href="#">Document</a> non nul et n'est pas <a href="#">inerte</a> .</b>	Le <a href="#">Document</a> pour lequel la <a href="#">fenêtre</a> a été créée.
Le contenu d'un <a href="#">iframe</a> .	
<b>Tout autre élément ou partie d'un élément déterminé par l'agent utilisateur comme étant une zone focalisable, en particulier pour faciliter l'accessibilité ou pour mieux correspondre aux conventions de la plate-forme.</b>	L'élément.
<p>Un agent utilisateur pourrait rendre toutes les puces des éléments de la liste <a href="#">séquentiellement focalisables</a> , afin qu'un utilisateur puisse naviguer plus facilement dans les listes.</p> <p>De même, un agent utilisateur pourrait rendre tous les éléments dotés <a href="#">title</a>d'attributs <a href="#">séquentiellement focalisables</a> , de sorte que leurs informations consultatives soient accessibles.</p>	
<i>Un <a href="#">conteneur navigable</a> (par exemple un <a href="#">iframe</a>) est une <a href="#">zone focalisable</a> , mais les événements clés acheminés vers un <a href="#">conteneur navigable</a> sont immédiatement acheminés vers <a href="#">le document actif</a> de son <a href="#">contenu navigable</a> . De même, dans la</i>	

*navigation à focus séquentiel, un conteneur navigable agit essentiellement comme un espace réservé pour le document actif de son contenu navigable .*

---

Une zone focalisable dans chacun Document est désignée comme **zone focalisée du document** . Le contrôle ainsi désigné change avec le temps, sur la base des algorithmes de cette spécification.

*Même si un document n'est pas entièrement actif et n'est pas affiché à l'utilisateur, il peut toujours avoir une zone ciblée du document . Si l'état entièrement actif d'un document change, sa zone ciblée du document restera la même.*

La zone actuellement focalisée d'un *traversable* **traversable de niveau supérieur** est la zone focalisable -ou-null renvoyée par cet algorithme :

1. Si *traversable* n'a pas le focus système , renvoie null.
2. Soit *candidat* le document actif traversable .
3. Alors que la zone ciblée du *candidat* est un conteneur navigable avec un contenu navigable non nul : définissez *le candidat* sur le document actif du contenu navigable de ce conteneur navigable .
4. Si la zone ciblée du *candidat* n'est pas nulle, définissez *le candidat* sur la zone ciblée du *candidat* .
5. *Candidat* de retour .

La chaîne de focus actuelle d'un *traversable de niveau supérieur* est la chaîne de focus de la zone actuellement focalisée de *traversable* , si *traversable* est non nul, ou une liste vide dans le cas contraire.

On dit qu'un élément qui est l' ancrage DOM d'une zone focalisable **obtient le focus** lorsque cette zone focalisable devient la zone actuellement focalisée d'un traversable de niveau supérieur . Lorsqu'un élément est l' ancrage DOM d'une zone focalisable de la zone actuellement focalisée d'un traversable de niveau supérieur , il est **focalisé** .

La **chaîne de mise au point** d'un *sujet* à zone de mise au point est la liste ordonnée construite comme suit :

1. Soit *output* une liste vide .
2. Laissez *currentObject* être *sujet* .
3. Alors que c'est vrai :

1. Ajoutez *currentObject* à *output* .
2. Si *currentObject* est area la forme d'un élément, ajoutez cet area élément à *output* .

Sinon, si l'ancree DOM de *currentObject* est un élément qui n'est pas *currentObject* lui-même, alors ajoutez l'ancree DOM de *currentObject* à *output* .

3. Si *currentObject* est une zone focalisable , alors définissez *currentObject* sur le document de noeud de l' ancree DOM de *currentObject* .

Sinon, si *currentObject* est un Document dont le parent du noeud navigable est non nul, alors définissez *currentObject* sur le parent du noeud navigable de *currentObject* .

Sinon, cassez .

4. *Sortie de retour* .

*La chaîne commence par le sujet et (si le sujet est ou peut être la zone actuellement ciblée d'un traversable de niveau supérieur ) continue dans la hiérarchie de mise au point jusqu'au traversable Document de niveau supérieur .*

Tous les éléments qui sont des zones focalisables sont dits **focalisables** .

Il existe deux types spéciaux de focalisation pour les zones focalisables :

- Une zone pouvant être focalisée est dite **séquentiellement focalisable** si elle est incluse dans son Document ordre de navigation de focalisation séquentielle et si l'agent utilisateur détermine qu'elle est séquentiellement focalisable.
- Une zone focalisable est dite **focalisable par clic** si l'agent utilisateur détermine qu'elle est focalisable par clic. Les agents utilisateurs doivent considérer les zones focalisables avec des valeurs tabindex non nulles comme pouvant être focalisées par clic.

*Les éléments qui ne sont pas focalisables ne sont pas des zones focalisables , et donc ne peuvent pas être focalisées séquentiellement et ne peuvent pas être focalisées par clic .*

*Être focalisable est une déclaration indiquant si un élément peut être focalisé par programme, par exemple via la focus() méthode ou autofocus l'attribut. En revanche, les focalisations séquentielles et les clics régissent la manière dont l'agent utilisateur répond à l'interaction de l'utilisateur : respectivement, à la navigation séquentielle et en tant que comportement d'activation .*

*L'agent utilisateur peut déterminer qu'un élément n'est pas focalisable séquentiellement même s'il est focalisable et est inclus dans son Document ordre de*

navigation focalisé séquentiel , selon les préférences de l'utilisateur. Par exemple, les utilisateurs de macOS peuvent configurer l'agent utilisateur pour qu'il ignore les éléments de contrôle hors formulaire, ou peuvent ignorer les liens lorsqu'ils effectuent une navigation séquentielle avec le focus uniquement avec la `Tab` touche (au lieu d'utiliser à la fois les touches `Option` et `Tab`).

De même, l'agent utilisateur peut déterminer qu'un élément n'est pas focusable par clic même s'il est focusable . Par exemple, dans certains agents utilisateurs, cliquer sur un contrôle de formulaire non modifiable ne le focalise pas, c'est-à-dire que l'agent utilisateur a déterminé que ces contrôles ne sont pas focalisables par clic.

Ainsi, un élément peut être focusable , mais ni focusable séquentiellement ni click focusable . Par exemple, dans certains agents utilisateurs, un contrôle de formulaire non modifiable avec une valeur tabindex entière négative ne serait pas focalisable via l'interaction de l'utilisateur, uniquement via des API programmatiques.

Lorsqu'un utilisateur active une zone focalisable par clic , l'agent utilisateur doit exécuter les étapes de focalisation sur la zone focalisable avec le déclencheur de focalisation défini sur " ".`click`

Notez que la focalisation n'est pas un comportement d'activation , c'est-à-dire que l'appel de la `click()` méthode sur un élément ou l'envoi d'un `click` événement synthétique sur celui-ci ne provoquera pas la focalisation de l'élément.

---

Un nœud est un **propriétaire d'étendue de navigation ciblée** s'il s'agit d'un Document, d'un hôte fantôme , d'un emplacement ou d'un élément dans l'état d'affichage du popover qui a également un invocateur de popover défini.

Chaque propriétaire d'étendue de navigation ciblée possède une **étendue de navigation ciblée** , qui est une liste d'éléments. Son contenu est déterminé comme suit :

Chaque élément *element* a un **propriétaire de navigation de focus associé** , qui est soit null , soit un propriétaire de portée de navigation de focus . Il est déterminé par l'algorithme suivant :

1. Si le parent de l' *élément* est null, alors retournez null.
2. Si le parent de l' *élément* est un hôte fantôme , alors renvoie l'emplacement assigné à l'élément .
3. Si le parent de l' *élément* est une racine fantôme , alors retournez l' hôte du parent .

4. Si le parent de l' *élément* est l' élément document , alors renvoie le nœud document du parent .
5. Si l'*élément* est dans l' état d'affichage du popover et qu'un invocateur de popover est défini, alors renvoie *element* .
6. Renvoie le propriétaire de navigation du focus associé au parent de l'*élément* .

Ensuite, le contenu d'une portée de navigation ciblée du *propriétaire* de la portée de navigation du focus est tous les éléments dont le propriétaire de la navigation du focus associé est *owner* .

*L'ordre des éléments dans une portée de navigation de focus n'affecte aucun des algorithmes de cette spécification. L'ordre ne devient important que pour les concepts d'étendue de navigation de mise au point ordonnée par index de tabulation et d'étendue de navigation de mise au point aplati par index de tabulation définis ci-dessous.*

Une **étendue de navigation de focus ordonnée par tabindex** est une liste de zones pouvant être ciblées et de propriétaires d'étendue de navigation de focus . Chaque *propriétaire* propriétaire de la portée de navigation du focus possède une portée de navigation du focus ordonnée par tabindex , dont le contenu est déterminé comme suit :

- Il contient tous les éléments de la portée de navigation du focus du *propriétaire* qui sont eux-mêmes propriétaires de la portée de navigation du focus , à l'exception des éléments dont la valeur tabindex est un entier négatif.
- Il contient toutes les zones focalisables dont l'ancre DOM est un élément dans la portée de navigation du focus du *propriétaire* , à l'exception des zones focalisables dont la valeur tabindex est un entier négatif.

L'ordre à l'intérieur d'une portée de navigation ciblée par tabindex est déterminé par la valeur tabindex de chaque élément , comme décrit dans la section ci-dessous.

*Les règles ne donnent pas un ordre précis, car elles sont principalement composées d'énoncés "devrait" et d'ordres relatifs.*

Une **étendue de navigation de mise au point ordonnée par tabindex aplati** est une liste de zones pouvant être ciblées . Chaque *propriétaire* propriétaire de la portée de navigation du focus possède une portée de navigation du focus ordonnée par tabindex aplati distincte , dont le contenu est déterminé par l'algorithme suivant :

1. Soit *le résultat* un clone de la portée de navigation du focus ordonné par tabindex du *propriétaire* .
2. Pour chaque *élément de résultat* :

1. Si l'élément n'est pas un [propriétaire d'étendue de navigation ciblée](#) , [continuez](#) .
2. Si *item* n'est pas une [zone focalisable](#) , remplacez *item* par tous les éléments de l' [étendue de navigation de focus aplati tabindex-ordered](#) de *item* .
3. Sinon, insérez le contenu de la portée de navigation du focus [aplati tabindex-ordered](#) item après *item* .

### 6.6.3 L' [tabindex](#)attribut



L' [tabindex](#) attribut content permet aux auteurs de faire d'un élément et des régions qui ont l'élément comme [ancres DOM](#) des [zones focalisables](#) , d'autoriser ou d'empêcher qu'elles soient [séquentiellement focalisables](#) et de déterminer leur ordre relatif pour [la navigation séquentielle](#) .

Le nom "index de tabulation" vient de l'utilisation courante de la [Tab](#) touche pour naviguer dans les éléments pouvant être sélectionnés. Le terme « tabulation » fait référence à l'avancement dans [des zones focalisables séquentiellement focalisables](#) .

L' [tabindex](#) attribut, s'il est spécifié, doit avoir une valeur qui est un [entier valide](#) . Les nombres positifs spécifient la position relative des [zones focalisables](#) de l'élément dans l' [ordre de navigation du focus séquentiel](#) , et les nombres négatifs indiquent que le contrôle n'est pas [focalisable séquentiellement](#) .

Les développeurs doivent faire preuve de prudence lorsqu'ils utilisent des valeurs autres que 0 ou -1 pour leurs [tabindex](#) attributs, car cela est compliqué à faire correctement.

*Ce qui suit fournit un résumé non normatif des comportements des [tabindex](#) valeurs d'attribut possibles. Le modèle de traitement ci-dessous donne les règles les plus précises.*

#### **omis (ou valeurs non entières)**

*L'agent utilisateur décidera si l'élément est [focusable](#) , et si c'est le cas, s'il est [focusable séquentiellement](#) ou [click focusable](#) (ou les deux).*

#### **-1 (ou autres valeurs entières négatives)**

*Permet à l'élément d'être [focalisable](#) et indique que l'auteur préférerait que l'élément soit [focalisable par clic](#) mais pas [séquentiellement focalisable](#) . L'agent utilisateur peut ignorer cette préférence pour le clic et la focalisation séquentielle, par exemple, pour des types d'éléments spécifiques*



*selon les conventions de la plate-forme, ou pour les utilisateurs utilisant uniquement le clavier.*

**0**

*Permet à l'élément d'être [focalisable](#) et indique que l'auteur préférerait que l'élément soit à la fois [focalisable par clic](#) et [focalisable séquentiellement](#) . L'agent utilisateur peut ignorer cette préférence pour le clic et la focalisation séquentielle.*

### **valeurs entières positives**

*Se comporte de la même manière que 0, mais crée en outre un ordre relatif dans une [étendue de navigation de focus ordonnée par tabindex](#) , de sorte que les éléments avec [tabindex](#) une valeur d'attribut plus élevée viennent plus tard.*

*Notez que l' [tabindex](#) attribut ne peut pas être utilisé pour rendre un élément non focalisable. La seule façon pour un auteur de page de le faire est de [désactiver](#) l'élément ou de le rendre [inerte](#) .*

---

La **valeur tabindex** d'un élément est la valeur de son [tabindex](#) attribut, analysée à l'aide des [règles d'analyse des entiers](#) . Si l'analyse échoue ou si l'attribut n'est pas spécifié, la [valeur tabindex](#) est nulle.

La [valeur tabindex](#) d'une [zone focalisable](#) est la [valeur tabindex](#) de son [ancree DOM](#) .

La [valeur tabindex](#) d'un élément doit être interprétée comme suit :

#### **Si la valeur est nulle**

L'agent utilisateur doit suivre les conventions de la plate-forme pour déterminer si l'élément doit être considéré comme une [zone focalisable](#) et si oui, si l'élément et toutes les [zones focalisables](#) qui ont l'élément comme [ancree DOM](#) sont [séquentiellement focalisables](#) , et si oui, quelle est leur position relative dans leur [portée de navigation de mise au point ordonnée par tabindex](#) doit être. Si l'élément est [propriétaire d'une portée de navigation ciblée](#) , il doit être inclus dans sa [portée de navigation ciblée classée par tabindex](#) même s'il ne s'agit pas d'une [zone pouvant être sélectionnée](#) .

L'ordre relatif dans une [portée de navigation de focus ordonnée par tabindex](#) pour les éléments et [les zones pouvant être sélectionnées](#) qui appartiennent à la même [portée de navigation de focus](#) et dont [la valeur tabindex](#) est null doit être dans [l'ordre de l'arborescence shadow-inclusive](#) .

Selon les conventions de la plate-forme modulo, il est suggéré que les éléments suivants soient considérés comme [des domaines focalisables](#) et soient [focalisables séquentiellement](#) :

- [a](#) les éléments qui ont un [href](#) attribut
- [button](#) éléments
- [input](#) les éléments dont [type](#) l'attribut n'est pas à l'état [Masqué](#)
- [select](#) éléments
- [textarea](#) éléments
- [summary](#) les éléments qui sont le premier [summary](#) élément enfant d'un [details](#) élément
- Éléments avec un [draggable](#) ensemble d'attributs, si cela permet à l'agent utilisateur d'autoriser l'utilisateur à commencer des opérations de glissement pour ces éléments sans utiliser de dispositif de pointage
- [Modification des hôtes](#)
- [Conteneurs navigables](#)

#### Si la valeur est un entier négatif

L'agent utilisateur doit considérer l'élément comme une [zone focalisable](#) , mais doit omettre l'élément de toute [portée de navigation de mise au point ordonnée par tabindex](#) .

*Une raison valable d'ignorer l'exigence selon laquelle la navigation par focus séquentiel ne permet pas à l'auteur d'accéder à l'élément serait si le seul mécanisme de l'utilisateur pour déplacer le focus est la navigation par focus séquentiel. Par exemple, un utilisateur au clavier uniquement ne pourrait pas cliquer sur un contrôle de texte avec un négatif [tabindex](#), de sorte que l'agent utilisateur de l'utilisateur serait bien justifié d'autoriser l'utilisateur à tabuler sur le contrôle malgré tout.*

#### Si la valeur est un zéro

L'agent utilisateur doit permettre à l'élément d'être considéré comme une [zone focalisable](#) et doit permettre à l'élément et à toutes les [zones focalisables](#) qui ont l'élément comme [ancres DOM](#) d'être [séquentiellement focalisables](#) .

L'ordre relatif au sein d'une [étendue de navigation de focus ordonnée par tabindex](#) pour les éléments et [les zones pouvant être sélectionnées](#) qui appartiennent à la même [étendue de navigation de focus](#) et dont [la valeur tabindex](#) est zéro doit être dans [l'ordre de l'arborescence shadow-inclusive](#) .

#### Si la valeur est supérieure à zéro



L'agent utilisateur doit permettre à l'élément d'être considéré comme une [zone focalisable](#) et doit permettre à l'élément et à toutes les [zones focalisables](#) qui ont l'élément comme [ancree DOM](#) d'être [focalisables de manière séquentielle](#) , et doit placer l'élément - référencé comme *candidat* ci-dessous - et les éléments susmentionnés [les zones focalisables](#) dans l' [étendue de navigation focalisée tabindex](#) dont l'élément fait partie de sorte que, par rapport aux autres éléments et [zones focalisables](#) qui appartiennent à la même [étendue de navigation focalisée](#) , elles sont :

- avant toute [zone focusable](#) dont l'[ancree DOM](#) est un élément dont [tabindex](#) l'attribut a été omis ou dont la valeur, une fois analysée, renvoie une erreur,
- avant toute [zone focusable](#) dont l'[ancree DOM](#) est un élément dont [tabindex](#) l'attribut a une valeur égale ou inférieure à zéro,
- après toute [zone focalisable](#) dont l'[ancree DOM](#) est un élément dont [tabindex](#) l'attribut a une valeur supérieure à zéro mais inférieure à la valeur de l' [tabindex](#) attribut sur *candidate* ,
- après toute [zone focalisable](#) dont l'[ancree DOM](#) est un élément dont [tabindex](#) l'attribut a une valeur égale à la valeur de l' [tabindex](#) attribut sur *candidat* mais qui est situé plus tôt que *candidat* dans l'[ordre de l'arborescence shadow-inclusive](#) ,
- avant toute [zone focalisable](#) dont l'[ancree DOM](#) est un élément dont [tabindex](#) l'attribut a une valeur égale à la valeur de l' [tabindex](#) attribut sur *le candidat* mais qui est situé après *le candidat* dans l'[ordre de l'arborescence incluant l'ombre](#) , et
- avant toute [zone focalisable](#) dont l'[ancree DOM](#) est un élément dont [tabindex](#) l'attribut a une valeur supérieure à la valeur de l' [tabindex](#) attribut sur *candidate* .



L' [tabIndex](#) attribut IDL doit [refléter](#) la valeur de l' [tabindex](#) attribut content. La valeur par défaut est 0 si l'élément est un élément [a](#), [area](#), [button](#), [frame](#), [iframe](#), [input](#), [object](#), [select](#), [textarea](#), ou [SVG a](#) , ou est un [summary](#) élément qui est un [résumé de ses détails parents](#) . La valeur par défaut est -1 sinon.

*La valeur par défaut variable en fonction du type d'élément est un artefact historique.*

## 6.6.4 Modèle de traitement

Pour **obtenir la zone focalisable** d'une *cible de focus* qui est soit un élément qui n'est pas une [zone focalisable](#) , soit un élément [navigable](#) , étant donné [un déclencheur de focus](#) de chaîne facultatif (par défaut " `other`"), exécutez le premier ensemble d'étapes correspondant dans la liste suivante :

**Si la cible de focus est un [area](#)élément avec une ou plusieurs formes qui sont [des zones focalisables](#)**

Renvoie la forme correspondant au premier [img](#)élément dans [l'arborescence](#) qui utilise l'image cliquable à laquelle [area](#)appartient l'élément.

**Si la cible de focus est un élément avec une ou plusieurs régions défilantes qui sont [des zones focalisables](#)**

Renvoie la première région défilable de l'élément, selon une précommande, parcours en profondeur de l' [arbre plat](#) . [\[CSSSCOPAGE\]](#)

**Si la cible du focus est l' [élément de document](#) de son [Document](#)**

Renvoie la [fenêtreDocument](#) de .

**Si la cible du focus est une [zone navigable](#)**

Retourne le [document actif](#) du [navigable](#) .

**Si la cible du focus est un [conteneur navigable](#) avec un contenu non nul [navigable](#)**

Renvoie le [document actif](#) du [contenu navigable](#) du [conteneur navigable](#) .

**Si la cible du focus est un [hôte fantôme](#) dont [le focus des délégués](#) de [la racine fantôme](#) est vrai**

1. Soit *focusElement* la zone actuellement focalisée de [l'ancre DOM](#) d' [un traversable de niveau supérieur](#) .
2. Si la cible du focus est un ancêtre [inclusif incluant shadow](#) de *focusElement* , alors retourne *focusElement* .
3. Renvoie le [délégué de focus](#) pour la cible de focus donnée *focus trigger* .

*Pour [la focalisation séquentielle](#) , la gestion du [focus des hôtes fantômes](#) et des délégués est effectuée lors de la construction de l' [ordre de navigation du focus séquentiel](#) . C'est-à-dire que les [étapes de mise au point](#) ne seront jamais appelées sur de tels [hôtes fantômes](#) dans le cadre de la navigation de mise au point séquentielle.*

**Sinon**

Renvoie nul.

Le **délégué de focus** pour un *focusTarget* , étant donné une chaîne facultative *focusTrigger* (par défaut " `other`") et un booléen facultatif *autofocusOnly* (par défaut `false`), est donné par les étapes suivantes :

1. Si *focusTarget* est un [hôte fantôme](#) et que [le focus](#) des délégués de sa [racine fantôme](#) est faux, alors renvoie null.
2. Soit *whereToLook* être *focusTarget*.
3. Si *whereToLook* est un [hôte fantôme](#), définissez *whereToLook* sur [la racine fantôme](#) de *whereToLook*.
4. Laissez *autofocusDelegate* être le [délégué de mise au point automatique](#) pour *whereToLook* étant donné *focusTrigger*.
5. Si *autofocusDelegate* n'est pas nul, alors retournez *autofocusDelegate*.
6. Si *autofocusOnly* est vrai, alors renvoie null.
7. [Pour chaque](#) descendant des [descendants](#) de *whereToLook*, dans [l'ordre arborescent](#) :

1. Laissez *focusableArea* être nul.
2. Si *focusTarget* est un [dialog](#) élément et descendant est [séquentiellement focalisable](#), alors définissez *focusableArea* sur descendant.
3. Sinon, si *focusTarget* n'est pas a [dialog](#) et descendant est une [zone focalisable](#), définissez *focusableArea* sur descendant.
4. Sinon, définissez *focusableArea* sur le résultat de [l'obtention de la zone focalisable](#) pour le descendant donné *focusTrigger*.

*Cette étape peut finir par se répéter, c'est-à-dire que les étapes [d'obtention de la zone focalisable](#) peuvent renvoyer le [délégué de focus](#) de descendant.*

5. Si *focusableArea* n'est pas null, alors retournez *focusableArea*.

*Il est important que nous ne regardions pas l'ombre [, y compris les descendants](#) ici, mais uniquement les [descendants](#). [Les hôtes fantômes](#) sont plutôt gérés par le cas récursif mentionné ci-dessus.*

8. Renvoie nul.

*L'algorithme ci-dessus renvoie essentiellement la première [zone focalisable](#) appropriée où le chemin entre son [ancrage DOM](#) et les délégués *focusTarget* se concentre sur toutes les limites de l'arbre fantôme.*

Le **délégué de mise au point automatique** pour une *cible de mise au point* donnée à un déclencheur de mise au point est donné par les étapes suivantes :

1. Pour chaque [descendant](#) descendant de *focus cible*, dans [l'ordre de l'arborescence](#) :

1. Si *descendant* n'a pas d' [autofocus](#) attribut de contenu, alors [continuez](#) .
  2. Soit *zone focalisable* être *descendant* , si *descendant* est une [zone focalisable](#) ; sinon, laissez *la zone focalisable* être le résultat de [l'obtention de la zone focalisable](#) pour *le déclencheur de focus* donné *descendant* .
  3. Si *la zone focalisable* est nulle, alors [continuez](#) .
  4. Si *la zone focalisable* n'est pas [focalisable par clic](#) et que *le déclencheur de focalisation* est " [click](#)", alors [continuez](#) .
  5. *Zone focalisable* de retour .
2. Renvoi nul.

Les **étapes de mise au point** pour une *nouvelle cible de mise au point* d'un objet qui est soit une [zone de mise au point](#) , soit un élément qui n'est pas une [zone de mise au point](#) , soit une [zone navigable](#) , sont les suivantes. Ils peuvent éventuellement être exécutés avec une *cible de secours* et un *déclencheur de focus* de chaîne .

1. Si *la nouvelle cible de mise au point* n'est pas une [zone de mise au point](#) , définissez *la nouvelle cible de mise au point* sur le résultat de [l'obtention de la zone de mise au point](#) pour *la nouvelle cible de mise au point* , en fonction du *déclencheur de mise au point* si elle a été transmise.
2. Si *la nouvelle cible de focus* est nulle, alors :
  1. Si aucune *cible de secours* n'a été spécifiée, alors retour.
  2. Sinon, définissez *la nouvelle cible de mise au point* sur la *cible de secours* .
3. Si *la nouvelle cible de focus* est un [conteneur navigable](#) avec un contenu non nul [navigable](#) , alors définissez *la nouvelle cible de focus* sur le [document actif](#) du [contenu navigable](#) .
4. Si *la nouvelle cible de focus* est une [zone focalisable](#) et que son [ancree DOM](#) est [inert](#) , alors retournez.
5. Si *la nouvelle cible de focus* est la [zone actuellement focalisée d'un traversable de niveau supérieur](#) , alors retournez.
6. Soit *l'ancienne chaîne* la [chaîne de focus actuelle de la traversée de niveau supérieur](#) dans laquelle se trouve *la nouvelle cible de focus* .
7. Soit *nouvelle chaîne* la [chaîne de focus](#) de *la nouvelle cible de focus* .
8. Exécutez les [étapes de mise à jour du focus](#) avec *l'ancienne chaîne* , *la nouvelle chaîne* et *la nouvelle cible de focus* respectivement.

Les agents utilisateurs doivent exécuter [immédiatement les étapes de focalisation](#) pour une [zone focalisable](#) ou un candidat [navigable](#) chaque fois que l'utilisateur tente de déplacer la focalisation sur *candidat* .

Les **étapes de défocalisation** pour une *ancienne cible de focus* d'objet qui est soit une [zone focalisable](#) , soit un élément qui n'est pas une [zone focalisable](#) sont les suivantes :

1. Si *l'ancienne cible de focus* est un [hôte fantôme](#) dont le focus des [délégués de la racine fantôme](#) est vrai, et que [la racine fantôme](#) de *l'ancienne cible de focus* est un [ancêtre incluant l'ombre](#) de la zone actuellement ciblée de l'ancre DOM d'un [traversable de niveau supérieur](#) , alors définir *l'ancienne cible de focus* sur la [zone actuellement ciblée d'un traversable de niveau supérieur](#) .
2. Si *l'ancienne cible de mise au point* est [inerte](#) , alors revenez.
3. Si *l'ancienne cible de focus* est un [area](#) élément et que l'une de ses formes est la [zone actuellement focalisée d'un traversable de niveau supérieur](#) , ou, si *l'ancienne cible de focus* est un élément avec une ou plusieurs régions défilables, et que l'une d'elles est la [zone actuellement focalisée de un traversable de niveau supérieur](#) , puis laissez *l'ancienne cible* être la [zone actuellement ciblée d'un traversable de niveau supérieur](#) .
4. Soit *l'ancienne chaîne* la [chaîne de focus actuelle de la traversée de niveau supérieur](#) dans laquelle se trouve *l'ancienne cible de focus* .
5. Si *l'ancienne cible de focus* n'est pas l'une des entrées de *l'ancienne chaîne* , alors revenez.
6. Si *l'ancienne cible de mise au point* n'est pas une [zone focalisable](#) , revenez.
7. Soit *topDocument* la dernière entrée de *l'ancienne chaîne* .
8. Si [le nœud navigable](#) de *topDocument* a [le focus système](#) , alors exécutez les [étapes de focus](#) pour [la fenêtre d'affichage](#) de *topDocument* .

Sinon, appliquez toutes les conventions pertinentes spécifiques à la plateforme pour supprimer [le focus système](#) du [nœud navigable](#) de *topDocument* et exécutez les [étapes de mise à jour du focus](#) en fonction de *l'ancienne chaîne* , d'une liste vide et de null.

*Les [étapes de non mise au point](#) n'entraînent pas toujours le changement de mise au point, même lorsqu'elles sont appliquées à la [zone actuellement mise au point d'un traversable de niveau supérieur](#) . Par exemple, si la [zone actuellement focalisée d'un traversable de niveau supérieur](#) est une [fenêtre](#) , elle conservera généralement son focus jusqu'à ce qu'une autre [zone focalisable](#) soit explicitement focalisée avec les [étapes de focalisation](#) .*

---

Les **étapes de mise à jour du focus** , étant donné respectivement une *ancienne chaîne* , une *nouvelle chaîne* et une *nouvelle cible de focus* , sont les suivantes :

1. Si la dernière entrée de *l'ancienne chaîne* et la dernière entrée de *la nouvelle chaîne* sont identiques, pop la dernière entrée de *l'ancienne chaîne* et la dernière entrée de *la nouvelle chaîne* et refaites cette étape.
2. Pour chaque *entrée* entry dans *old chain* , dans l'ordre, exécutez ces sous-étapes :

1. Si *l'entrée* est un input élément, et que l' change événement s'applique à l'élément, et que l'élément n'a pas de comportement d'activation défini , et que l'utilisateur a modifié la valeur de l'élément ou sa liste de fichiers sélectionnés alors que le contrôle était focalisé sans valider ce changement (comme qu'il est différent de ce qu'il était lorsque le contrôle a été focalisé pour la première fois), puis déclenchez un événement nommé change au niveau de l'élément, avec l' bubbles attribut initialisé à true.

2. Si *entry* est un élément, laissez *blur event target* être *entry* .

Si *entry* est un Document objet, laissez *blur event target* être l' objet global pertinent Document de cet objet .

Sinon, laissez *la cible de l'événement de flou* être nulle.

3. Si *l'entrée* est la dernière entrée de *l'ancienne chaîne* et que *l'entrée* est un Element, et que la dernière entrée de *la nouvelle chaîne* est également un Element, alors laissez *la cible de flou* associée être la dernière entrée de *la nouvelle chaîne* . Sinon, laissez *la cible de flou* associée être nulle.
4. Si *la cible de l'événement de flou* n'est pas nulle, déclenchez un événement de mise au point nommé blur à *la cible de l'événement de flou* , avec *la cible de flou* associée comme cible associée.

*Dans certains cas, par exemple, si entry est area la forme d'un élément, une région déroulante ou un viewport , aucun événement n'est déclenché.*

3. Appliquez toutes les conventions spécifiques à la plate-forme pour cibler *la nouvelle cible de focus* . (Par exemple, certaines plates-formes sélectionnent le contenu d'un contrôle de texte lorsque ce contrôle a le focus.)
4. Pour chaque *entrée* entry in *new chain* , dans l'ordre inverse, exécutez ces sous-étapes :
  1. Si *l'entrée* est une zone focalisable : désignez *l'entrée* comme zone focalisée du document .
  2. Si *entry* est un élément, laissez *focus event target* être *entry* .



Si *entry* est un [Document](#) objet, laissez *focus event target* être l' [objet global pertinent](#) [Document](#) de cet objet .

Sinon, laissez *la cible de l'événement de focus* être nulle.

3. Si *l'entrée* est la dernière entrée dans *la nouvelle chaîne* et que *l'entrée* est un [Element](#), et que la dernière entrée dans *l'ancienne chaîne* est également un [Element](#), alors laissez *la cible de focus associée* être la dernière entrée dans *l'ancienne chaîne* . Sinon, laissez *la cible de focus associée* être nulle.
4. Si *la cible de l'événement de focus* n'est pas nulle, [déclenchez un événement de focus](#) nommé [focus](#) sur *la cible de l'événement de focus* , avec *la cible de focus associée* comme cible associée.

*Dans certains cas, par exemple si *entry* est [area](#) la forme d'un élément, une région déroulante ou un [viewport](#) , aucun événement n'est déclenché.*

Pour **déclencher un événement de focus** nommé *e* sur un élément *t* avec une cible connexe donnée *r* , [déclenchez un événement](#) nommé *e* sur *t* , en utilisant [FocusEvent](#), avec l' [relatedTarget](#) attribut initialisé à *r* , l' [view](#) attribut initialisé à l'[objet global pertinent](#) du [document de nœud](#) de *t* , et le jeu [de drapeaux composé](#) .

---

Lorsqu'un événement clé doit être acheminé dans un [traversable de niveau supérieur](#) , l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit *la zone cible* la [zone actuellement focalisée du traversable de niveau supérieur](#) .
2. [Assert](#) : *la zone cible* n'est pas nulle, car les événements clés ne sont acheminés que vers [les traversables de niveau supérieur](#) qui ont [le focus système](#) . Par conséquent, *la zone cible* est une [zone focalisable](#) .
3. Soit *le nœud cible* l' [ancree DOM](#) de *la zone cible* .
4. Si *le nœud cible* est un [Document](#) qui a un [élément de corps](#) , alors laissez *le nœud cible* être [l'élément de corps](#) de cela [Document](#).

Sinon, si *le nœud cible* est un objet qui a un [élément de document](#) [Document](#) non nul , alors laissez *le nœud cible* être cet [élément de document](#) .

5. Si *le nœud cible* n'est pas [inert](#) , alors :

1. Soit *canHandle* le résultat de [la distribution](#) de l'événement key au *nœud cible* .
  2. Si *canHandle* est vrai, alors laissez *la zone cible* gérer l'événement clé. Cela peut inclure [le déclenchement d'un \*click\* événement](#) sur *le nœud cible* .
- 

Les **étapes has focus** , étant donné un [Document](#) objet *target* , sont les suivantes :

1. Si le traversable de [niveau supérieur du nœud navigable](#) de *la cible* n'a pas [le focus système](#) , alors renvoie false.
2. Soit *candidate* le document *actif* du [nœud navigable](#) de la cible navigable de [niveau supérieur](#) .
3. Alors que c'est vrai :
  1. Si *candidate* est *target* , alors retourne true.
  2. Si la [zone focalisée](#) de *candidate* est un [conteneur navigable](#) avec un contenu non nul [navigable](#) , alors définissez *candidate* sur le [document actif du contenu](#) navigable de ce [conteneur navigable](#) .
  3. Sinon, renvoie faux.

### 6.6.5 Navigation de mise au point séquentielle

Chacun [Document](#) a un **ordre de navigation de mise au point séquentielle** , qui ordonne certaines ou toutes les [zones focalisables](#) les unes par [Document](#) rapport aux autres. Son contenu et son ordre sont donnés par l' [étendue de navigation de mise au point ordonnée par tabindex aplati](#) du fichier [Document](#).

*Selon les règles définissant la [portée de navigation du focus aplati tabindex-ordered focus](#) , l'ordre n'est pas nécessairement lié à l' [ordre](#) de l'arborescence du [Document](#).*

Si une [zone focalisable](#) est omise de l' [ordre de navigation de mise au point séquentielle](#) de son [Document](#), elle est alors inaccessible via [la navigation de mise au point séquentielle](#) .



Il peut également y avoir un **point de départ de navigation séquentielle** . Il est initialement non défini. L'agent utilisateur peut le définir lorsque l'utilisateur indique qu'il doit être déplacé.

Par exemple, l'agent utilisateur pourrait le définir à la position du clic de l'utilisateur si l'utilisateur clique sur le contenu du document.

*Les agents utilisateurs doivent définir le point de départ de la navigation séquentielle sur l' élément cible lors de la navigation vers un fragment .*

Lorsque l'utilisateur demande que le focus se déplace de la zone actuellement ciblée d'un niveau supérieur traversable vers la zone ciblée suivante ou précédente (par exemple, comme action par défaut consistant à appuyer sur la `tab` touche), ou lorsque l'utilisateur demande que le focus se déplace séquentiellement vers un niveau supérieur . accessible en premier lieu (par exemple depuis la barre d'adresse du navigateur), l'agent utilisateur doit utiliser l'algorithme suivant :

1. Soit le point de départ soit la zone actuellement focalisée d'un traversable de niveau supérieur , si l'utilisateur a demandé de déplacer le focus séquentiellement à partir de là, ou bien le traversable de niveau supérieur lui-même, si l'utilisateur a plutôt demandé de déplacer le focus depuis l'extérieur du traversable de niveau supérieur .
2. S'il existe un point de départ de navigation de mise au point séquentielle défini et qu'il se trouve à l'intérieur du point de départ , laissez alors point de départ être le point de départ de la navigation de mise au point séquentielle à la place.
3. Laissez la direction être vers l'avant si l'utilisateur a demandé le contrôle suivant , et vers l'arrière si l'utilisateur a demandé le contrôle précédent.

*Généralement, appuyer sur `tab` demande la commande suivante et appuyer sur `shift + tab` demande la commande précédente.*

4. Boucle : Laissez le mécanisme de sélection être séquentiel si le point de départ est un navigable ou si le point de départ est dans son Document ordre séquentiel de navigation .

Sinon, le point de départ n'est pas dans son Document ordre de navigation séquentiel ; laissez le mécanisme de sélection être DOM .

5. Soit *candidat* le résultat de l'exécution de l' algorithme de recherche par navigation séquentielle avec point de départ , direction et mécanisme de sélection comme arguments.
6. Si le candidat n'est pas nul, exécutez les étapes de focalisation pour le candidat et le retour.
7. Sinon, annulez le point de départ de la navigation avec mise au point séquentielle .

8. Si le point de départ est un [traversable de niveau supérieur](#) ou une [zone focalisable](#) dans le [traversable de niveau supérieur](#) , l'agent utilisateur doit transférer le focus sur ses propres contrôles de manière appropriée (le cas échéant), en respectant *la direction* , puis revenir.

Par exemple, si *direction* est *back* , alors le dernier contrôle [séquentiellement focalisable](#) avant la zone de rendu du navigateur serait le contrôle à focaliser.

Si l'agent utilisateur n'a pas de contrôles [séquentiellement focalisables](#) - un navigateur en mode kiosque, par exemple - alors l'agent utilisateur peut à la place redémarrer ces étapes avec le *point de départ* étant le [traversable de niveau supérieur](#) lui-même.

9. Sinon, le point de départ est une [zone focalisable](#) dans un [enfant navigable](#) . Définissez le point de départ sur le [parent](#) de cet [élément navigable enfant](#) et revenez à l'étape intitulée *boucle* .

L' **algorithme de recherche par navigation séquentielle** comprend les étapes suivantes. Cet algorithme prend trois arguments : *point de départ* , *direction* et *mécanisme de sélection* .

1. Sélectionnez la cellule appropriée dans le tableau suivant et suivez les instructions de cette cellule.

La cellule appropriée est celle qui provient de la colonne dont l'en-tête décrit *la direction* et de la première ligne dont l'en-tête décrit *le point de départ* et le *mécanisme de sélection* .

	<i>la direction est vers l'avant</i>	<i>la direction est en arrière</i>
<i>le point de départ est une <a href="#">voie navigable</a></i>	Soit <i>candidate</i> la première <a href="#">zone pouvant être mise au point séquentiellement appropriée</a> dans le <a href="#">document actif</a> du point de départ , le cas échéant ; sinon nul	Soit <i>candidate</i> la dernière <a href="#">zone pouvant être mise au point séquentiellement appropriée</a> dans le <a href="#">document actif</a> du point de départ , le cas échéant ; sinon nul
<i>le mécanisme de sélection est DOM</i>	Soit <i>candidate</i> la première <a href="#">zone pouvant être mise au point séquentiellement appropriée</a> dans le <a href="#">document d'origine</a> suivant le point de départ , le cas échéant ; sinon nul	Soit <i>candidate</i> la dernière <a href="#">zone pouvant être mise au point séquentiellement appropriée</a> dans le <a href="#">document personnel</a> précédant le point de départ , le cas échéant ; sinon nul
<i>le mécanisme de sélection est séquentiel</i>	Soit <i>candidate</i> la première <a href="#">zone pouvant être mise au point séquentiellement appropriée</a> dans l' <a href="#">ordre de</a>	Soit <i>candidate</i> la dernière <a href="#">zone pouvant être mise au point séquentiellement appropriée</a> dans l' <a href="#">ordre de</a>

	<i>la direction est vers l'avant</i>	<i>la direction est en arrière</i>
	<a href="#">navigation de mise au point séquentielle</a> de départ suivant <i>le point de départ</i> , le cas échéant ; sinon nul	<a href="#">navigation de mise au point séquentielle</a> de départ précédant <i>le point de départ</i> , le cas échéant ; sinon nul

Une **zone focalisable séquentiellement appropriée** est une [zone focalisable](#) dont [l'ancre DOM](#) n'est pas [inerte](#) et est [focalisable séquentiellement](#) .

Le **document d'origine** est le [Document](#) auquel appartient *le point de départ* .

L' **ordre de navigation de mise au point séquentielle de départ** est l' [ordre de navigation de mise au point séquentielle](#) auquel appartient *le point de départ* .

*L' [ordre de navigation de mise au point séquentielle d'accueil](#) est l' [ordre de navigation de mise au point séquentielle](#) du [document d'accueil](#) , mais n'est utilisé que lorsque le point de départ se trouve dans cet [ordre de navigation de mise au point séquentielle](#) (quand ce n'est pas le cas, le mécanisme de sélection sera DOM ).*

- Si *candidat* est un [conteneur navigable avec un contenu](#) non nul navigable , alors laissez *new candidate* être le résultat de l'exécution de [l'algorithme de recherche de navigation séquentielle](#) avec [le contenu navigable](#) du *candidat* comme premier argument, *direction* comme deuxième et *séquentiel* comme troisième.

Si *nouveau candidat* est nul, laissez *le point de départ* être *candidat* , et revenez au début de cet algorithme. Sinon, laissez *le candidat* être *nouveau candidat* .

- Candidat* de retour .

### 6.6.6 API de gestion des focus

```
dictionary FocusOptions {
```

```
  boolean preventScroll = false;
```

```
  boolean focusVisible;
```

```
};
```

`documentOrShadowRoot.activeElement`



Renvoie l'élément le plus profond du document par lequel ou vers lequel les événements clés sont acheminés. C'est, grosso modo, l'élément central du document.

Pour les besoins de cette API, lorsqu'un [objet navigable enfant](#) est ciblé, son [conteneur](#) est [ciblé](#) dans [le document actif](#) de son [parent](#) . Par exemple, si l'utilisateur déplace le focus vers un contrôle de texte dans un , est l'élément renvoyé par l' API dans le [nœud document](#) de [.iframeiframeactiveElementiframe](#)

De même, lorsque l'élément focalisé se trouve dans une [arborescence de nœuds](#) différente de *documentOrShadowRoot* , l'élément renvoyé sera l' [hôte](#) situé dans la même [arborescence de nœuds](#) que *documentOrShadowRoot* si *documentOrShadowRoot* est un [ancêtre incluant l'ombre](#) de l'élément focalisé, et null sinon.

`document.hasFocus()`

✓

Renvoie true si les événements clés sont acheminés via ou vers le document ; sinon, renvoie faux. Grosso modo, cela correspond au document, ou à un document imbriqué dans celui-ci, qui est mis au point.

`window.focus()`

✓

[Déplace le focus vers le navigable](#) de la fenêtre , le cas échéant.

`element.focus([ { preventScroll: true } ])`

✓

Déplace le focus sur l'élément.

Si l'élément est un [conteneur navigable](#) , déplace le focus sur son [contenu navigable](#) à la place.

Par défaut, cette méthode fait également défiler l'élément dans la vue. Fournir l' [preventScroll](#) option et la définir sur true empêche ce comportement.

`element.blur()`

✓

Déplace le focus vers la [fenêtre](#) . L'utilisation de cette méthode est déconseillée ; si vous voulez focaliser la [fenêtre](#) , appelez la [focus\(\)](#) méthode sur l' [élément documentDocument](#) de .

N'utilisez pas cette méthode pour masquer la bague de mise au point si vous trouvez la bague de mise au point disgracieuse. Au lieu de cela, utilisez la [:focus-visible](#) pseudo-classe pour remplacer la propriété ['outline'](#) et fournir une manière différente de montrer quel élément est ciblé. Sachez que si un style de mise au point alternatif n'est pas disponible, la page sera beaucoup moins utilisable pour les personnes qui naviguent principalement sur les pages à l'aide d'un clavier, ou pour les personnes ayant une vision

réduite qui utilisent des contours de mise au point pour les aider à naviguer dans la page.

Par exemple, pour masquer le contour des `textarea` éléments et utiliser à la place un arrière-plan jaune pour indiquer le focus, vous pouvez utiliser :

```
textarea:focus-visible { outline: none; background: yellow;
color: black; }
```

Le `activeElement` getter de l'attribut doit exécuter ces étapes :

1. Soit *candidat* l' [ancree DOM](#) de la [zone focalisée](#) de ce [document](#) `DocumentOrShadowRoot` de nœud .
2. Définissez le *candidat* sur le résultat du [recyclage](#) du *candidat* par rapport à this `DocumentOrShadowRoot`.
3. Si [la racine](#) du *candidat* n'est pas this , alors retourne `null`. `DocumentOrShadowRoot`
4. Si *candidat* n'est pas un `Document` objet, alors renvoie *candidat* .
5. Si le *candidat* a un [élément body](#) , alors retournez cet [élément body](#) .
6. Si l'[élément de document](#) du *candidat* n'est pas nul, alors retournez cet [élément de document](#) .
7. Renvoie nul.

La `hasFocus()` méthode sur l' `Document` objet, lorsqu'elle est invoquée, doit renvoyer le résultat de l'exécution des [étapes has focus](#) avec l' `Document` objet comme argument.

La `focus()` méthode, lorsqu'elle est invoquée, doit exécuter ces étapes :

1. Soit *courant* le [navigable](#) de cet `Window` objet .
2. Si *courant* est nul, alors retour.
3. Exécutez les [étapes de mise au point](#) avec le *courant* .
4. Si *current* est un [traversable de niveau supérieur](#) , les agents utilisateurs sont encouragés à déclencher une sorte de notification pour indiquer à l'utilisateur que la page tente d'obtenir le focus.



Les `blur()` étapes de la méthode consistent à ne rien faire.

*Historiquement, les méthodes `focus()` et `blur()` affectaient en fait le focus au niveau du système du widget système (par exemple, onglet ou fenêtre) qui contenait le [navigable](#), mais les sites hostiles abusent largement de ce comportement au détriment de l'utilisateur.*

La méthode sur les éléments, lorsqu'elle est invoquée, doit exécuter les étapes suivantes : `focus(options)`

1. Si l'élément est marqué comme [verrouillé pour le focus](#), alors revenez.
2. Marquez l'élément comme **verrouillé pour le focus**.
3. Exécutez les [étapes de mise au point](#) pour l'élément.
4. Si la valeur du `focusVisible` membre du dictionnaire d' `options` est true, ou n'est pas présente mais d'une manière [définie par l'implémentation](#), l'agent utilisateur détermine qu'il serait préférable de le faire, alors [indiquez focus](#).
5. Si la valeur du `preventScroll` membre du dictionnaire d' `options` est false, faites [défiler l'élément dans la vue](#) avec le comportement de défilement " `auto`", la position de direction de flux de bloc définie sur une valeur [définie par l'implémentation](#) et la position de direction de base en ligne définie sur une valeur [définie par l'implémentation](#).
6. Décochez l'élément comme [verrouillé pour le focus](#).

La `blur()` méthode, lorsqu'elle est invoquée, doit exécuter les [étapes de désactivation](#) pour l'élément sur lequel la méthode a été appelée. Les agents utilisateurs peuvent ignorer sélectivement ou uniformément les appels à cette méthode pour des raisons d'utilisabilité.

Par exemple, si la `blur()` méthode est utilisée de manière imprudente pour supprimer la bague de mise au point pour des raisons esthétiques, la page deviendrait inutilisable par les utilisateurs du clavier. Ignorer les appels à cette méthode permettrait ainsi aux utilisateurs du clavier d'interagir avec la page.

### 6.6.7 L' `autofocus` attribut

L' `autofocus` attribut content permet à l'auteur d'indiquer qu'un élément doit être focalisé dès que la page est chargée, permettant à l'utilisateur de commencer à taper sans avoir à focaliser manuellement l'élément principal.

Lorsque l' `autofocus` attribut est spécifié sur un élément à l'intérieur `dialog` d'éléments ou [d'éléments HTML](#) dont `popover` l'attribut est défini, il sera mis en évidence lorsque la boîte de dialogue ou le popover s'affichera.

L' `autofocus` attribut est un [attribut booléen](#).

Pour trouver l' **élément racine de portée autofocus ancêtre le plus proche** étant donné un Element *élément* :

1. Si *element* est un dialogélément, alors retourne *element* .
2. Si l'attribut de l' *élément*popover n'est pas dans l' état sans popover , alors retourne *element* .
3. Soit *ancêtre* élément . \_
4. Alors que *ancestor* a un élément parent :
  1. Définissez *ancêtre* sur l'élément parent de *l'ancêtre* .
  2. Si *ancestor* est un dialogélément, alors retourne *ancestor* .
  3. Si l' attribut *ancestor*popover n'est pas dans l' état no popover , alors renvoie *ancestor* .
5. Retour *ancêtre* .

Il ne doit pas y avoir deux éléments avec le même élément racine de portée autofocus ancêtre le plus proche qui ont tous deux l' autofocusattribut spécifié.

Chacun Document a une liste de candidats autofocus , initialement vide.

Each Document has an **autofocus processed flag** boolean, initially false.

When an element with the autofocus attribute specified is inserted into a document, run the following steps:

1. If the user has indicated (for example, by starting to type in a form control) that they do not wish focus to be changed, then optionally return.
2. Let *target* be the element's node document.
3. If *target* is not fully active, then return.
4. If *target*'s active sandboxing flag set has the sandboxed automatic features browsing context flag, then return.
5. Pour chaque *ancestorNavigable* des navigables ancêtres de *la cible* :  
si l'origine du document actif de *ancestorNavigable* n'est pas la même origine que l'origine de *la cible* , alors retour.
6. Supposons que *topDocument* soit le document actif du nœud navigable de la cible navigable de niveau supérieur .
7. Si l'indicateur de traitement de mise au point automatique de *topDocument* est faux, supprimez l'élément des candidats de mise au point automatique de



*topDocument* et [ajoutez](#) l'élément aux [candidats de mise au point automatique](#) de *topDocument* .

*Nous ne vérifions pas si un élément est une [zone focalisable](#) avant de le stocker dans la liste [des candidats autofocus](#) , car même s'il ne s'agit pas d'une zone focalisable lors de son insertion, il pourrait le devenir au moment où [les candidats autofocus flush](#) le verront.*

Pour **vider les candidats de mise au point automatique** pour un document *topDocument* , exécutez ces étapes :

1. Si [l'indicateur de traitement de mise au point automatique](#) de *topDocument* est vrai, alors retournez.
2. Laissez *les candidats* être [les candidats autofocus](#) de *topDocument* .
3. Si *candidats* [est vide](#) , alors retour.
4. Si [la zone ciblée](#) de *topDocument* n'est pas *topDocument* lui-même, ou si *topDocument* a [un élément cible](#) non nul , alors :
  1. *Candidats* [vides](#) .
  2. Définissez [l'indicateur de traitement de mise au point automatique](#) de *topDocument* sur true.
  3. Retour.
5. Tant que *candidats* n'est pas [vide](#) :
  1. Soit *element* les *candidats* [0].
  2. Soit *doc* le [nœud document](#) de l' élément .
  3. Si *doc* n'est pas [entièrement actif](#) , [supprimez](#) l'élément des *candidats* et [continuez](#) .
  4. Si [le nœud navigable](#) de [niveau supérieur](#) de *doc* n'est pas le même que [le nœud navigable](#) de *topDocument* , [supprimez](#) l'élément des *candidats* et [continuez](#) .
  5. Si [le jeu de feuilles de style de blocage de script](#) de *doc* n'est pas [vide](#) , alors retournez.

*Dans ce cas, element est actuellement le meilleur candidat, mais doc n'est pas prêt pour la mise au point automatique. Nous réessayerons la prochaine fois que [les candidats à l'autofocus flush](#) seront appelés.*

6. [Supprimer](#) l'élément des *candidats* .



7. Soit *inclusiveAncestorDocuments* une [liste](#) composée du [document actif](#) des [navigables ancêtres inclusifs](#) de *doc* .
8. Si l'un [Document](#) des éléments *inclusAncestorDocuments* a [un élément cible](#) non nul , [continuez](#) .
9. Soit *cible* élément . \_
10. Si *la cible* n'est pas une [zone focalisable](#) , définissez *la cible* sur le résultat de [l'obtention de la zone focalisable](#) pour *la cible* .

*Les candidats à la mise au point automatique peuvent [contenir des éléments qui ne sont pas des zones focalisables](#) . En plus des cas particuliers traités dans l' algorithme [d'obtention de la zone focalisable](#) , cela peut se produire parce qu'un élément [de zone non focalisable](#) [autofocus](#) avec un attribut a été [inséré dans un document](#) et qu'il n'est jamais devenu focalisable, ou parce que l'élément était focalisable mais son statut a changé pendant que il a été stocké dans [les candidats autofocus](#) .*

11. Si *la cible* n'est pas nulle, alors :
  1. *Candidats* [vides](#) .
  2. Définissez [l'indicateur de traitement de mise au point automatique](#) de *topDocument* sur true.
  3. Exécutez les [étapes de mise au point](#) pour *la cible* .

*Ceci gère la mise au point automatique pendant le chargement du document. Les méthodes [show\(\)](#) et des éléments traitent également l' attribut [showModal\(\)](#) [dialogautofocus](#)*  
*La focalisation de l'élément n'implique pas que l'agent utilisateur doive focaliser la fenêtre du navigateur si elle a perdu le focus.*



L' [autofocus](#) attribut IDL doit [refléter](#) l'attribut de contenu du même nom.

Dans l'extrait de code suivant, le contrôle de texte serait ciblé lors du chargement du document.

```
<input maxlength="256" name="q" value="" autofocus>
<input type="submit" value="Search">
```

L' [autofocus](#) attribut s'applique à tous les éléments, pas seulement aux contrôles de formulaire. Cela permet des exemples tels que les suivants :

```
<div contenteditable autofocus>Edit <strong>me!</strong></div>
```

## 6.7 Attribuer des raccourcis clavier

### 6.7.1 Présentation

*Cette section est non normative.*

Chaque élément pouvant être activé ou ciblé peut se voir attribuer une seule combinaison de touches pour l'activer, à l'aide de l' `accesskey` attribut.

Le raccourci exact est déterminé par l'agent utilisateur, en fonction des informations sur le clavier de l'utilisateur, des raccourcis clavier qui existent déjà sur la plate-forme et des autres raccourcis spécifiés sur la page, en utilisant les informations fournies dans l'attribut comme guide `accesskey`.

Afin de s'assurer qu'un raccourci clavier pertinent est disponible sur une grande variété de périphériques d'entrée, l'auteur peut fournir un certain nombre d'alternatives dans l' `accesskey` attribut.

Chaque alternative se compose d'un seul caractère, tel qu'une lettre ou un chiffre.

Les agents utilisateurs peuvent fournir aux utilisateurs une liste des raccourcis clavier, mais les auteurs sont encouragés à le faire également. L' `accessKeyLabel` attribut IDL renvoie une chaîne représentant la combinaison de touches réelle attribuée par l'agent utilisateur.

Dans cet exemple, un auteur a fourni un bouton qui peut être appelé à l'aide d'une touche de raccourci. Pour prendre en charge les claviers complets, l'auteur a fourni "C" comme clé possible. Pour prendre en charge les appareils équipés uniquement de pavés numériques, l'auteur a prévu "1" comme autre touche possible.

```
<input type=button value=Collect onclick="collect()"
      accesskey="C 1" id=c>
```

Pour indiquer à l'utilisateur quelle est la touche de raccourci, l'auteur a choisi ce script ici pour ajouter explicitement la combinaison de touches à l'étiquette du bouton :

```
function addShortcutKeyLabel(button) {
    if (button.accessKeyLabel != '')
        button.value += ' (' + button.accessKeyLabel + ')';
}
addShortcutKeyLabel(document.getElementById('c'));
```

Les navigateurs sur différentes plates-formes afficheront différentes étiquettes, même pour la même combinaison de touches, en fonction de la convention qui prévaut sur cette plate-forme. Par exemple, si la combinaison de touches est la touche Contrôle, la touche Maj et la lettre C, un navigateur Windows peut afficher "Ctrl+Maj+C", alors qu'un navigateur Mac peut afficher "⌘C", alors qu'un

navigateur Emacs pourrait simplement afficher "CC". De même, si la combinaison de touches est la touche Alt et la touche Échap, Windows peut utiliser "Alt+Échap", Mac pourrait utiliser "⌘", et un navigateur Emacs pourrait utiliser "M-ESC" ou "ESC".

En général, par conséquent, il n'est pas judicieux d'essayer d'analyser la valeur renvoyée par l' `accessKeyLabel` attribut IDL.

### 6.7.2 L' `accesskey` attribut



Tous [les éléments HTML](#) peuvent avoir l' `accesskey` attribut content défini. La `accesskey` valeur de l'attribut est utilisée par l'agent utilisateur comme guide pour créer un raccourci clavier qui active ou focalise l'élément.

Si spécifié, la valeur doit être un [ensemble ordonné de jetons uniques séparés par des espaces](#) dont aucun n'est [identique à](#) un autre jeton et dont chacun doit avoir exactement un point de code de longueur.

Dans l'exemple suivant, une variété de liens sont donnés avec des touches d'accès afin que les utilisateurs du clavier familiarisés avec le site puissent naviguer plus rapidement vers les pages pertinentes :

```
<nav>
  <p>
    <a title="Consortium Activities" accesskey="A"
href="/Consortium/activities">Activities</a> |
    <a title="Technical Reports and Recommendations" accesskey="T"
href="/TR/">Technical Reports</a> |
    <a title="Alphabetical Site Index" accesskey="S"
href="/Consortium/siteindex">Site Index</a> |
    <a title="About This Site" accesskey="B"
href="/Consortium/">About Consortium</a> |
    <a title="Contact Consortium" accesskey="C"
href="/Consortium/contact">Contact</a>
  </p>
</nav>
```

Dans l'exemple suivant, le champ de recherche reçoit deux clés d'accès possibles, "s" et "0" (dans cet ordre). Un agent utilisateur sur un appareil avec un clavier complet peut choisir comme touche de raccourci, tandis qu'un agent utilisateur sur un petit appareil avec juste un pavé numérique peut choisir juste la touche simple sans fioritures :Ctrl + Alt + S0

```
<form action="/search">
```

```

<label>Search: <input type="search" name="q" accesskey="s
0"></label>

<input type="submit">

</form>

```

Dans l'exemple suivant, un bouton a des clés d'accès possibles décrites. Un script essaie ensuite de mettre à jour l'étiquette du bouton pour annoncer la combinaison de touches sélectionnée par l'agent utilisateur.

```

<input type=submit accesskey="N @ 1" value="Compose">
...
<script>
function labelButton(button) {
    if (button.accessKeyLabel)
        button.value += ' (' + button.accessKeyLabel + ')';
}
var inputs = document.getElementsByTagName('input');
for (var i = 0; i < inputs.length; i += 1) {
    if (inputs[i].type == "submit")
        labelButton(inputs[i]);
}
</script>

```

Sur un agent utilisateur, l'étiquette du bouton peut devenir "Composer (⌘N)". Sur un autre, ça pourrait devenir "Composer (Alt+␣+1)". Si l'agent utilisateur n'attribue pas de clé, ce sera juste "Composer". La chaîne exacte dépend de la [clé d'accès attribuée](#) et de la manière dont l'agent utilisateur représente cette combinaison de clés.

### 6.7.3 Modèle de traitement

La **clé d'accès attribuée** à un élément est une combinaison de clés dérivée de l'[accesskey](#) attribut de contenu de l'élément. Initialement, un élément ne doit pas avoir de [clé d'accès attribuée](#).

Chaque fois que l'attribut d'un élément [accesskey](#) est défini, modifié ou supprimé, l'agent utilisateur doit mettre à jour la [clé d'accès attribuée](#) à l'élément en exécutant les étapes suivantes :

1. Si l'élément n'a pas [accesskey](#) d'attribut, passez à l'étape *de secours* ci-dessous.
2. Sinon, [divisez la valeur de l'attribut sur l'espace blanc ASCII](#) et laissez les clés être les jetons résultants.

3. Pour chaque valeur dans *les clés* à tour de rôle, dans l'ordre dans lequel les jetons sont apparus dans la valeur de l'attribut, exécutez les sous-étapes suivantes :
  1. Si la valeur n'est pas une chaîne d'exactly un point de code de longueur, ignorez le reste de ces étapes pour cette valeur.
  2. Si la valeur ne correspond pas à une touche du clavier du système, ignorez le reste de ces étapes pour cette valeur.
  3. Si l'agent utilisateur peut trouver une combinaison de zéro ou plusieurs touches de modification qui, combinées à la clé qui correspond à la valeur donnée dans l'attribut, peuvent être utilisées comme clé d'accès, alors l'agent utilisateur peut attribuer cette combinaison de touches comme [clé d'accès attribuée](#) à l'élément et retour.
  4. *Fallback* : Facultativement, l'agent utilisateur peut attribuer une combinaison de touches de son choix comme [clé d'accès attribuée](#) à l'élément , puis revenir.
  5. Si cette étape est atteinte, l'élément n'a pas [de clé d'accès attribuée](#) .

Une fois qu'un agent utilisateur a sélectionné et assigné une clé d'accès pour un élément, l'agent utilisateur ne devrait pas changer la [clé d'accès assignée](#) à l'élément à moins que l' [accesskey](#)attribut content soit changé ou que l'élément soit déplacé vers un autre [Document](#).

Lorsque l'utilisateur appuie sur la combinaison de touches correspondant à la [touche d'accès attribuée](#) à un élément, si l'élément [définit une commande](#) , la facette [Etat masqué](#) de la commande est fausse (visible), la facette [Etat désactivé](#) de la commande est également fausse (activée), l'élément est [dans un document qui a un contexte de navigation](#) non nul , et ni l'élément ni aucun de ses ancêtres n'a d' [hidden](#)attribut spécifié, alors l'agent utilisateur doit déclencher l' [Action](#) de la commande.

*Les agents utilisateurs [peuvent également exposer](#) des éléments qui ont un [accesskey](#)attribut d'autres manières, par exemple dans un menu affiché en réponse à une combinaison de touches spécifique.*

L' **accessKey**attribut IDL doit [réfléter](#) l' **accesskey**attribut content.

MDN

L' **accessKeyLabel**attribut IDL doit renvoyer une chaîne qui représente la [clé d'accès affectée](#) à l'élément , le cas échéant. Si l'élément n'en a pas, alors l'attribut IDL doit renvoyer la chaîne vide.

## 6.8 Édition

### 6.8.1 Rendre les régions de document modifiables : l' **contenteditable**attribut de contenu

✓ MDN

```
interface mixin ElementContentEditable {  
  
    [CEReactions] attribute DOMString contentEditable;  
  
    [CEReactions] attribute DOMString enterKeyHint;  
  
    readonly attribute boolean isContentEditable;  
  
    [CEReactions] attribute DOMString inputMode;  
  
};
```

✓ MDN

L' **contenteditable**attribut content est un [attribut énuméré](#) dont les mots clés sont la chaîne vide, `true` et `false`. La chaîne vide et le `true` mot-clé correspondent à l'état *vrai* . Le `false` mot-clé correspond à l'état *faux* . De plus, il existe un troisième état, l'état *d'héritage* , qui est la [valeur manquante default](#) et la [valeur invalide default](#) .

L'état *vrai* indique que l'élément est modifiable. L'état *d'héritage* indique que l'élément est modifiable si son parent l'est. L'état *faux* indique que l'élément n'est pas modifiable.

Par exemple, considérez une page qui a un **form** et un **textarea** pour publier un nouvel article, où l'utilisateur est censé écrire l'article en HTML :

```
<form method=POST>  
  <fieldset>  
    <legend>New article</legend>
```

```

    <textarea name=article>&lt;p>Hello world.&lt;/p></textarea>
  </fieldset>
  <p><button>Publish</button></p>
</form>

```

Lorsque le script est activé, l' [textarea](#) élément peut être remplacé par un contrôle de texte enrichi à la place, en utilisant l' [contenteditable](#) attribut :

```

<form method=POST>
  <fieldset>
    <legend>New article</legend>
    <textarea id=textarea name=article>&lt;p>Hello
world.&lt;/p></textarea>
    <div id=div style="white-space: pre-wrap" hidden><p>Hello
world.</p></div>
    <script>
      let textarea = document.getElementById("textarea");
      let div = document.getElementById("div");
      textarea.hidden = true;
      div.hidden = false;
      div.contentEditable = "true";
      div.oninput = (e) => {
        textarea.value = div.innerHTML;
      };
    </script>
  </fieldset>
  <p><button>Publish</button></p>
</form>

```

Les fonctionnalités à activer, par exemple l'insertion de liens, peuvent être implémentées à l'aide de l' [document.execCommand\(\)](#) API ou à l'aide [Selection](#) d'API et d'autres API DOM. [\[COMMANDEEXEC\]](#) [\[SELECTION\]](#) [\[DOM\]](#)

L' [contenteditable](#) attribut peut également être utilisé à bon escient :

```

<!doctype html>
<html lang=en>
<title>Live CSS editing!</title>
<style style=white-space:pre contenteditable>
html { margin:.2em; font-size:2em; color:lime; background:purple }
head, title, style { display:block }
body { display:none }
</style>

```

**element.contentEditable** [ = value ]

Renvoie " true", " false" ou " inherit", en fonction de l'état de l' contentEditableattribut.

Peut être défini, pour changer cet état.

Lève un "SyntaxError" DOMException si la nouvelle valeur n'est pas l'une de ces chaînes.

**element.isContentEditable**



Renvoie vrai si l'élément est modifiable ; sinon, renvoie faux.

L' **contentEditable**attribut IDL, lors de l'obtention, doit renvoyer la chaîne " true" si l'attribut de contenu est défini sur l'état vrai, " false" si l'attribut de contenu est défini sur l'état faux, et " inherit" sinon. Lors de la définition, si la nouvelle valeur est une correspondance ASCII insensible à la casse pour la chaîne " inherit", l'attribut de contenu doit être supprimé, si la nouvelle valeur est une correspondance ASCII insensible à la casse pour la chaîne " true", alors l'attribut de contenu doit être défini à la chaîne " true", si la nouvelle valeur est une correspondance ASCII insensible à la casse pour la chaîne " false", alors l'attribut de contenu doit être défini sur la chaîne " false", SyntaxError" DOMException .

L' **isContentEditable**attribut IDL, à l'obtention, doit retourner true si l'élément est soit un hôte d'édition soit editable , et false sinon.

## 6.8.2 Rendre des documents entiers modifiables : le designModegetter et le setter

**document.designMode** [ = value ]



Renvoie " on" si le document est éditable, et " off" s'il ne l'est pas.

Peut être défini pour modifier l'état actuel du document. Cela met le document au point et réinitialise la sélection dans ce document.

Documentles objets ont un **mode de conception associé activé** , qui est un booléen. C'est d'abord faux.

Les **designMode**étapes du getter consistent à renvoyer " on" si le mode de conception activé est vrai ; sinon " off".

Les designModeétapes du setter sont :

1. Soit *value* la valeur donnée, convertie en ASCII minuscule .



2. Si *la valeur* est " `on`" et [que le](#) mode de [conception activé](#) est faux, alors :
  1. Définissez [le](#) mode [de conception activé](#) sur `true`.
  2. Réinitialisez les points de délimitation de début et de fin de [cette plage active](#) pour qu'ils soient au début de [celle-ci](#) .
  3. Exécutez les [étapes de mise au point](#) pour [cet élément de document](#) , s'il n'est pas nul.
3. Si *la valeur* est " `off`", définissez [ce](#) mode de [conception activé](#) sur faux.

### 6.8.3 Meilleures pratiques pour les éditeurs de pages

Les auteurs sont encouragés à définir la propriété ['white-space'](#) sur [les hôtes d'édition](#) et sur le balisage créé à l'origine via ces mécanismes d'édition sur la valeur `'pre-wrap'`. La gestion des espaces blancs HTML par défaut n'est pas bien adaptée à l'édition WYSIWYG, et le retour à la ligne ne fonctionnera pas correctement dans certains cas extrêmes si ['white-space'](#) est laissé à sa valeur par défaut.

Comme exemple de problèmes qui surviennent si la valeur par défaut "normale" est utilisée à la place, considérons le cas où l'utilisateur tape " `yellow␣␣ball`", avec deux espaces (représentés ici par " `␣`" ) entre les mots. Avec les règles d'édition en place pour la valeur par défaut de ['white-space'](#) ("normal"), le balisage résultant consistera soit en "jaune `␣` balle" ou " `␣` balle jaune" ; c'est-à-dire qu'il y aura un espace insécable entre les deux mots en plus de l'espace régulier. Ceci est nécessaire car la valeur "normale" pour "espace blanc" [nécessite](#) que les espaces réguliers adjacents soient regroupés.

Dans le cas précédent, "jaune␣" peut passer à la ligne suivante ("␣" étant utilisé ici pour représenter un espace insécable) même si "jaune" seul peut tenir en bout de ligne ; dans ce dernier cas, "␣balle", s'il était enveloppé au début de la ligne, aurait une indentation visible à partir de l'espace insécable.

Lorsque ['white-space'](#) est défini sur `'pre-wrap'`, cependant, les règles d'édition mettront simplement deux espaces réguliers entre les mots, et si les deux mots sont séparés à la fin d'une ligne, les espaces seraient parfaitement supprimé du rendu.

### 6.8.4 Modification des API

Un **hôte d'édition** est soit un [élément HTML](#) avec son [contenteditable](#) attribut à l'état *vrai* , soit un [élément HTML enfant](#) d'un dont [le mode de conception activé](#) est vrai. [Document](#)

La définition des termes [plage active](#) , [hôte de modification de](#) et [modifiable](#) , les exigences d'interface utilisateur des éléments qui [modifient les hôtes](#) ou [modifiables](#) , les méthodes [execCommand\(\)](#) , [queryCommandEnabled\(\)](#) , [queryCommandIndeterm\(\)](#) , [queryCommandState\(\)](#) , [queryCommandSupported\(\)](#) et [queryCommandValue\(\)](#) , les sélections de texte et l' algorithme [de suppression de la sélection](#) sont définis dans `execCommand` . [\[EXECCOMMANDE\]](#)

## 6.8.5 Vérification orthographique et grammaticale

Les agents utilisateurs peuvent prendre en charge la vérification de l'orthographe et de la grammaire du texte modifiable, soit dans les contrôles de formulaire (comme la valeur des `textarea` éléments), soit dans les éléments d'un [hôte d'édition](#) (par exemple en utilisant [contenteditable](#)).

Pour chaque élément, les agents utilisateurs doivent établir un **comportement par défaut** , soit via des valeurs par défaut, soit via des préférences exprimées par l'utilisateur. Il existe trois comportements par défaut possibles pour chaque élément :

### ***vrai par défaut***

L'orthographe et la grammaire de l'élément seront vérifiées si son contenu est modifiable et si la vérification orthographique n'est pas explicitement désactivée via l' `spellcheck` attribut.

### ***faux par défaut***

L'orthographe et la grammaire de l'élément ne seront jamais vérifiées à moins que la vérification orthographique ne soit explicitement activée via l' `spellcheck` attribut.

### ***hériter par défaut***

Le comportement par défaut de l'élément est le même que celui de son élément parent. Les éléments qui n'ont pas d'élément parent ne peuvent pas avoir ceci comme comportement par défaut.



L' `spellcheck` attribut est un [attribut énuméré](#) dont les mots clés sont la chaîne vide `true` et `false`. La chaîne vide et le `true` mot-clé correspondent à l'état *vrai* . Le `false` mot-clé correspond à l'état *faux* . De plus, il existe un troisième

état, l'état *par défaut*, qui est la [valeur manquante default](#) et la [valeur invalide default](#).

*L'état vrai indique que l'orthographe et la grammaire de l'élément doivent être vérifiées. L'état par défaut indique que l'élément doit agir selon un comportement par défaut, éventuellement basé sur l'[spellcheck](#) état propre de l'élément parent, comme défini ci-dessous. L'état faux indique que l'élément n'est pas à vérifier.*

---

**`element.spellcheck [ = value ]`**

Renvoie true si l'orthographe et la grammaire de l'élément doivent être vérifiées ; sinon, renvoie faux.

Peut être défini pour remplacer la valeur par défaut et définir l'[spellcheck](#)attribut de contenu.

L'[spellcheck](#)attribut IDL, à l'obtention, doit retourner true si l'[spellcheck](#)attribut content de l'élément est dans l'état *true*, ou si l'[spellcheck](#)attribut content de l'élément est dans l'état *par défaut* et que le [comportement par défaut](#) de l'élément est [true-by-default](#), ou si l'[spellcheck](#)attribut content de l'élément est dans l'état *par défaut* et le [comportement par défaut](#) de l'élément est [hérité par défaut](#) et l'attribut IDL de l'élément parent de l'élément [spellcheck](#)renverrait vrai ; sinon, si aucune de ces conditions ne s'applique, l'attribut doit à la place renvoyer false.

*L'[spellcheck](#)attribut IDL n'est pas affecté par les préférences de l'utilisateur qui remplacent l'[spellcheck](#)attribut de contenu et peut donc ne pas refléter l'état réel de la vérification orthographique.*

Lors de la définition, si la nouvelle valeur est true, l'[spellcheck](#)attribut de contenu de l'élément doit être défini sur la chaîne littérale " `true`", sinon il doit être défini sur la chaîne littérale " `false`".

---

Les agents utilisateurs ne doivent considérer que les éléments de texte suivants comme vérifiables pour les besoins de cette fonctionnalité :

- La [valeur](#) des [input](#)éléments dont [type](#)les attributs sont dans les états [Text](#), [Search](#), [URL](#) ou [Email](#) et qui sont [modifiables](#) (c'est-à-dire qui n'ont pas l'[readonly](#)attribut spécifié et qui ne sont pas [désactivés](#)).

- La valeur des textarea éléments qui n'ont pas d' readonly attribut et qui ne sont pas désactivés .
- Texte dans Text les nœuds qui sont des enfants d' hôtes d'édition ou d'éléments modifiables .
- Texte dans les attributs des éléments modifiables .

Pour le texte qui fait partie d'un Text nœud, l'élément auquel le texte est associé est l'élément qui est le parent immédiat du premier caractère du mot, de la phrase ou de tout autre morceau de texte. Pour le texte dans les attributs, il s'agit de l'élément de l'attribut. Pour les valeurs des éléments input et textarea, il s'agit de l'élément lui-même.

Pour déterminer si un mot, une phrase ou un autre morceau de texte dans un élément applicable (tel que défini ci-dessus) doit avoir une vérification orthographique et grammaticale activée, l'UA doit utiliser l'algorithme suivant :

1. Si l'utilisateur a désactivé la vérification de ce texte, la vérification est désactivée.
2. Sinon, si l'utilisateur a forcé la vérification de ce texte à toujours être activée, alors la vérification est activée.
3. Sinon, si l'élément auquel le texte est associé a un spellcheck attribut de contenu, alors : si cet attribut est à l' état *vrai* , alors la vérification est activée ; sinon, si cet attribut est à l' état *faux* , la vérification est désactivée.
4. Sinon, s'il existe un élément ancêtre avec un spellcheck attribut de contenu qui n'est pas dans l' état *par défaut* , alors : si l'attribut de contenu de cet ancêtre le plus proche spellcheck est dans l' état *vrai* , alors la vérification est activée ; sinon, la vérification est désactivée.
5. Sinon, si le comportement par défaut de l'élément est true-by-default , la vérification est activée.
6. Sinon, si le comportement par défaut de l'élément est false-by-default , la vérification est désactivée.
7. Sinon, si l'élément parent de l'élément a sa vérification activée, alors la vérification est activée.
8. Sinon, la vérification est désactivée.

Si la vérification est activée pour un mot/une phrase/un texte, l'agent utilisateur doit indiquer les fautes d'orthographe et de grammaire dans ce texte. Les agents utilisateurs doivent tenir compte des autres sémantiques données dans le document lorsqu'ils suggèrent des corrections d'orthographe et de grammaire. Les agents utilisateurs peuvent utiliser la langue de l'élément pour déterminer les règles d'orthographe et de grammaire à utiliser, ou peuvent utiliser les paramètres de

langue préférés de l'utilisateur. Les UA devraient utiliser [input](#) des attributs d'élément tels que [pattern](#) pour s'assurer que la valeur résultante est valide, dans la mesure du possible.

Si la vérification est désactivée, l'agent utilisateur ne doit pas indiquer les fautes d'orthographe ou de grammaire pour ce texte.

L'élément avec l'ID "a" dans l'exemple suivant serait celui utilisé pour déterminer si le mot "Hello" est vérifié pour les fautes d'orthographe. Dans cet exemple, ce ne serait pas le cas.

```
<div contenteditable="true">
  <span spellcheck="false" id="a">Hell</span><em>o!</em>
</div>
```

L'élément avec l'ID "b" dans l'exemple suivant aurait la vérification activée (le caractère d'espace de début dans la valeur de l'attribut sur l' [input](#) élément entraîne l'ignorance de l'attribut, donc la valeur de l'ancêtre est utilisée à la place, quelle que soit la valeur par défaut).

```
<p spellcheck="true">
  <label>Name: <input spellcheck=" false" id="b"></label>
</p>
```

*Cette spécification ne définit pas l'interface utilisateur pour les vérificateurs d'orthographe et de grammaire. Un agent utilisateur peut proposer une vérification à la demande, effectuer une vérification continue pendant que la vérification est activée ou utiliser d'autres interfaces.*

## 6.8.6 Autocapitalisation

Certaines méthodes de saisie de texte, par exemple les claviers virtuels sur les appareils mobiles, ainsi que la saisie vocale, aident souvent les utilisateurs en mettant automatiquement en majuscule la première lettre des phrases (lors de la composition de texte dans une langue avec cette convention). Un clavier virtuel qui implémente la mise en majuscule automatique peut automatiquement basculer vers l'affichage des lettres majuscules (mais permettre à l'utilisateur de le remettre en minuscule) lorsqu'une lettre qui doit être mise en majuscule automatique est sur le point d'être tapée. D'autres types d'entrée, par exemple l'entrée vocale, peuvent effectuer une capitalisation automatique d'une manière qui ne donne pas aux utilisateurs la possibilité d'intervenir en premier. L' [autocapitalize](#) attribut permet aux auteurs de contrôler un tel comportement.

L' [autocapitalize](#) attribut, tel qu'il est généralement implémenté, n'affecte pas le comportement lors de la frappe sur un clavier physique. (Pour cette raison, ainsi que la possibilité pour les utilisateurs de remplacer le comportement de capitalisation

automatique dans certains cas ou de modifier le texte après la saisie initiale, l'attribut ne doit pas être utilisé pour toute sorte de validation de saisie.)

L' `autocapitalize` attribut peut être utilisé sur un [hôte d'édition](#) pour contrôler le comportement de mise en majuscule automatique pour la région modifiable hébergée, sur un élément `input` ou `textarea` pour contrôler le comportement de saisie de texte dans cet élément, ou sur un `form` élément pour contrôler le comportement par défaut de tous [les éléments héritant de la mise en majuscule automatique](#) associés avec l' `form` élément.

L' `autocapitalize` attribut n'entraîne jamais l'activation de la capitalisation automatique pour `input` les éléments dont `type` l'attribut est dans l'un des états [URL](#) , [Email](#) ou [Mot de passe](#) . (Ce comportement est inclus dans l' algorithme [d'indice de mise en majuscule automatique utilisé](#) ci-dessous.)

Le modèle de traitement de l'autocapitalisation est basé sur la sélection parmi cinq **indices d'autocapitalisation** , définis comme suit :

#### **défaut**

L'agent utilisateur et la méthode de saisie doivent déterminer eux-mêmes s'il faut ou non activer la capitalisation automatique.

#### **aucun**

Aucune capitalisation automatique ne doit être appliquée (toutes les lettres doivent être en minuscules par défaut).

#### **phrases**

La première lettre de chaque phrase doit être une majuscule par défaut ; toutes les autres lettres doivent être en minuscules par défaut.

#### **mots**

La première lettre de chaque mot doit être une majuscule par défaut ; toutes les autres lettres doivent être en minuscules par défaut.

#### **personnages**

Toutes les lettres doivent être en majuscules par défaut.



L' `autocapitalize` attribut est un [attribut énuméré](#) dont les états sont les [conseils d'autocapitalisation](#) possibles . L' [indicateur d'autocapitalisation](#) spécifié par l'état de l'attribut se combine avec d'autres considérations pour former l' [indicateur d'autocapitalisation utilisé](#) , qui informe le comportement de l'agent utilisateur. Les mots clés de cet attribut et leurs mappages d'état sont les suivants :

Mot-clé	État
<code>off</code>	

Mot-clé	État
<code>none</code>	<a href="#">aucun</a>
<code>on</code>	<a href="#">phrases</a>
<code>sentences</code>	
<code>words</code>	<a href="#">mots</a>
<code>characters</code>	<a href="#">personnages</a>

La valeur invalide par défaut est l' état [des phrases](#) . La valeur manquante par défaut est l' état [par défaut](#) .

`element.autocapitalize [ = value ]`

Renvoie l'état actuel de capitalisation automatique de l'élément, ou une chaîne vide s'il n'a pas été défini. Notez que pour les éléments `input` et `textarea` qui héritent de leur état d'un `form` élément, cela renverra l'état d'autocapitalisation de l' `form` élément, mais pour un élément dans une région modifiable, cela ne renverra pas l'état d'autocapitalisation de l'hôte d'édition (sauf si cet élément est, dans fait, l' [hôte d'édition](#) ).

Peut être défini pour définir l' `autocapitalize` attribut de contenu (et ainsi modifier le comportement de capitalisation automatique de l'élément).

Pour calculer le **propre indice d'autocapitalisation** d'un élément `element` , exécutez les étapes suivantes :

1. Si l' `autocapitalize` attribut content est présent sur `element` et que sa valeur n'est pas la chaîne vide, retournez l'état de l'attribut.
2. Si `element` est un [élément héritant de la mise en majuscule automatique et a un propriétaire de formulaire](#) non nul , renvoie le [propre indice d'autocapitalisation](#) du [propriétaire de formulaire](#) de l' `élément` .
3. Retour [par défaut](#) .

Les `autocapitalize` étapes du getter consistent à :

1. Soit `state` le [propre indice d'autocapitalisation](#) de [this](#) .
2. Si `state` est [default](#) , renvoie la chaîne vide.
3. Si `state` est [none](#) , alors retournez " `none` ".
4. Si `state` est [phrases](#) , alors retourne " `sentences` ".
5. Renvoie la valeur du mot-clé correspondant à `state` .

Les `autocapitalize` étapes du setter consistent à définir l' `autocapitalize` attribut content sur la valeur donnée.

---

Les agents utilisateurs qui prennent en charge le comportement de capitalisation automatique personnalisable pour une méthode de saisie de texte et souhaitent permettre aux développeurs Web de contrôler cette fonctionnalité doivent, lors de la saisie de texte dans un élément, calculer l' **indicateur de capitalisation automatique utilisé** pour l'élément. Il s'agira d'un [indice d'autocapitalisation](#) décrivant le comportement d'autocapitalisation recommandé pour la saisie de texte dans l'élément.

Les agents utilisateurs ou les méthodes d'entrée peuvent choisir d'ignorer ou de remplacer l' [indication de mise en majuscule automatique utilisée](#) dans certaines circonstances.

L' [indicateur d'autocapitalisation utilisé](#) pour un élément `element` est calculé à l'aide de l'algorithme suivant :

1. Si *element* est un [input](#) élément dont [type](#) l'attribut est dans l'un des états [URL](#) , [Email](#) ou [Mot de passe](#) , alors renvoie [default](#) .
2. Si *l'élément* est un [input](#) élément ou un [textarea](#) élément, alors renvoie [le propre indice d'autocapitalisation](#) de *l'élément* .
3. Si *element* est un [hôte d'édition](#) ou un élément [modifiable](#) , renvoie le [propre indice de mise en majuscule automatique](#) de l' [hôte d'édition de](#) *element* .
4. [Assert](#) : cette étape n'est jamais atteinte, car la saisie de texte n'a lieu que dans les éléments qui répondent à l'un des critères ci-dessus.

### 6.8.7 Modalités de saisie : l' [inputmode](#) attribut

Les agents utilisateurs peuvent prendre en charge l' [inputmode](#) attribut sur les contrôles de formulaire (comme la valeur des [textarea](#) éléments), ou dans les éléments d'un [hôte d'édition](#) (par exemple, en utilisant [contenteditable](#)).



L' [inputmode](#) attribut de contenu est un [attribut énuméré](#) qui spécifie le type de mécanisme de saisie qui serait le plus utile pour les utilisateurs saisissant du contenu.

Mot-clé	Description
<a href="#">none</a>	L'agent utilisateur ne doit pas afficher de clavier virtuel. Ce mot-clé est utile pour le contenu qui affiche son propre contrôle clavier.



Mot-clé	Description
<b>text</b>	L'agent utilisateur doit afficher un clavier virtuel capable de saisir du texte dans les paramètres régionaux de l'utilisateur.
<b>tel</b>	L'agent utilisateur doit afficher un clavier virtuel capable de saisir un numéro de téléphone. Cela devrait inclure les touches pour les chiffres de 0 à 9, le caractère "#" et le caractère "*". Dans certains pays, cela peut également inclure des étiquettes mnémoniques alphabétiques (par exemple, aux États-Unis, la clé étiquetée "2" est historiquement également étiquetée avec les lettres A, B et C).
<b>url</b>	L'agent utilisateur doit afficher un clavier virtuel capable de saisir du texte dans les paramètres régionaux de l'utilisateur, avec des touches pour faciliter la saisie des <a href="#">URL</a> , comme celle pour le "/" et "." caractères et pour une saisie rapide des chaînes que l'on trouve couramment dans les noms de domaine tels que "www." ou ".com".
<b>email</b>	L'agent utilisateur doit afficher un clavier virtuel capable de saisir du texte dans les paramètres régionaux de l'utilisateur, avec des touches pour faciliter la saisie des adresses e-mail, comme celle pour le caractère "@" et le "." personnage.
<b>numeric</b>	L'agent utilisateur doit afficher un clavier virtuel capable de saisie numérique. Ce mot-clé est utile pour la saisie du code PIN.
<b>decimal</b>	L'agent utilisateur doit afficher un clavier virtuel capable de saisie numérique fractionnaire. Les touches numériques et le séparateur de format pour les paramètres régionaux doivent être affichés.
<b>search</b>	L'agent utilisateur doit afficher un clavier virtuel optimisé pour la recherche.

L' **inputMode**attribut IDL doit [réfléter](#) l' **inputmode**attribut content, [limité aux seules valeurs connues](#) .

Lorsque **inputmode**n'est pas spécifié (ou est dans un état non pris en charge par l'agent utilisateur), l'agent utilisateur doit déterminer le clavier virtuel par défaut à afficher. Des informations contextuelles telles que l'entrée **type**ou **pattern**les attributs doivent être utilisées pour déterminer quel type de clavier virtuel doit être présenté à l'utilisateur.

### 6.8.8 Modalités de saisie : l' **enterkeyhint** attribut

Les agents utilisateurs peuvent prendre en charge l' **enterkeyhint** attribut sur les contrôles de formulaire (comme la valeur des **textarea**éléments), ou dans les éléments d'un [hôte d'édition](#) (par exemple, en utilisant **contenteditable**).



L' **enterkeyhint**attribut de contenu est un [attribut énuméré](#) qui spécifie l'étiquette d'action (ou l'icône) à présenter pour la touche Entrée sur les claviers virtuels. Cela

permet aux auteurs de personnaliser la présentation de la touche Entrée afin de la rendre plus utile aux utilisateurs.

Mot-clé	Description
<b>enter</b>	L'agent utilisateur doit présenter un signal pour l'opération « entrer », en insérant généralement une nouvelle ligne.
<b>done</b>	L'agent utilisateur doit présenter un signal pour l'opération "terminée", ce qui signifie généralement qu'il n'y a plus rien à saisir et que l'éditeur de méthode d'entrée (IME) sera fermé.
<b>go</b>	L'agent utilisateur doit présenter un signal pour l'opération « aller », ce qui signifie généralement amener l'utilisateur à la cible du texte qu'il a tapé.
<b>next</b>	L'agent utilisateur doit présenter un signal pour l'opération 'next', amenant généralement l'utilisateur au champ suivant qui acceptera du texte.
<b>previous</b>	L'agent utilisateur doit présenter un signal pour l'opération « précédent », amenant généralement l'utilisateur au champ précédent qui acceptera le texte.
<b>search</b>	L'agent utilisateur doit présenter un signal pour l'opération « recherche », amenant généralement l'utilisateur aux résultats de la recherche du texte qu'il a tapé.
<b>send</b>	L'agent utilisateur doit présenter un signal pour l'opération « envoyer », en livrant généralement le texte à sa cible.



L' **enterKeyHint**attribut IDL doit [refléter](#) l' **enterkeyhint**attribut content, [limité aux seules valeurs connues](#) .

Lorsque **enterkeyhint**n'est pas spécifié (ou est dans un état non pris en charge par l'agent utilisateur), l'agent utilisateur doit déterminer l'étiquette (ou l'icône) d'action par défaut à présenter. Des informations contextuelles telles que les attributs **inputmode**, **type**ou **pattern** doivent être utilisées pour déterminer l'étiquette d'action (ou l'icône) à afficher sur le clavier virtuel.

## 6.9 Rechercher dans la page

### 6.9.1 Présentation

Cette section définit **la recherche dans la page** - un mécanisme commun d'agent utilisateur qui permet aux utilisateurs de rechercher des informations particulières dans le contenu de la page.

L'accès à la fonctionnalité [de recherche dans la page](#) est fourni via une **interface de recherche dans la page** . Il s'agit d'une interface utilisateur fournie par l'agent utilisateur, qui permet à l'utilisateur de spécifier l'entrée et les paramètres de la

recherche. Cette interface peut apparaître à la suite d'un raccourci ou d'une sélection de menu.

Une combinaison de saisie de texte et de paramètres dans l' [interface de recherche dans la page](#) représente la **requête** de l'utilisateur . Cela inclut généralement le texte que l'utilisateur souhaite rechercher, ainsi que des paramètres facultatifs (par exemple, la possibilité de limiter la recherche à des mots entiers uniquement).

L'agent utilisateur traite le contenu de la page pour une [requête](#) donnée et identifie zéro ou plusieurs **correspondances** , qui sont des plages de contenu qui satisfont la [requête](#) de l'utilisateur .

L'une des [correspondances](#) est identifiée par l'utilisateur comme étant la **correspondance active** . Il est mis en surbrillance et défile dans la vue. L'utilisateur peut parcourir les [correspondances](#) en faisant avancer la [correspondance active](#) à l'aide de l' [interface de recherche dans la page](#) .

[Le numéro 3539](#) suit la standardisation de la façon dont [la recherche dans la page](#) sous-tend l'API actuellement non spécifiée `window.find()` .

### 6.9.2 Interaction avec [details](#) et [hidden=until-found](#)

Lorsque la recherche dans la page commence à rechercher des correspondances, tous [details](#) les éléments de la page dont l'attribut n'est pas [open](#) défini doivent voir le [contenu ignoré](#) de leur deuxième emplacement devenir accessible, sans modifier l' [open](#) attribut, afin de rendre la recherche dans la page capable pour y chercher. De même, tous les éléments HTML avec l' [hidden](#) attribut dans l' [état masqué jusqu'à ce qu'ils soient trouvés](#) doivent voir leur [contenu ignoré](#) devenir accessible sans modifier l' [hidden](#) attribut afin de rendre la recherche dans la page capable de les parcourir. Une fois que la recherche dans la page a terminé la recherche de correspondances, les [details](#) éléments et les éléments avec l' [hidden](#) attribut dans l' [état masqué jusqu'à la recherche](#) devraient voir leur contenu être à nouveau ignoré. L'ensemble de ce processus doit se dérouler de manière synchrone (et n'est donc pas observable pour les utilisateurs ou pour l'auteur du code). [\[CSSCONTAIN\]](#)

Lorsque find-in-page choisit une nouvelle [correspondance active](#) , procédez comme suit :

1. Soit *node* le premier nœud de la [correspondance active](#) .
2. [Mettez en file d'attente une tâche globale](#) sur la [source de tâche d'interaction utilisateur](#) en fonction de [l'objet global pertinent](#) du *nœud* pour exécuter les étapes suivantes :
  1. Exécutez l' [algorithme révélant les détails de l'ancêtre](#) sur le *nœud* .

2. Exécutez l' [algorithme de révélation de l'ancêtre caché jusqu'à ce qu'il soit trouvé](#) sur le *nœud* .

*Lorsque [find-in-page](#) développe automatiquement un [details](#)élément comme celui-ci, il déclenche un [toggle](#)événement. Comme avec l' [scroll](#)événement distinct qui déclenche la recherche dans la page, cet événement peut être utilisé par la page pour découvrir ce que l'utilisateur tape dans la boîte de dialogue de recherche dans la page. Si la page crée une petite zone de défilement avec le terme de recherche actuel et chaque caractère suivant possible que l'utilisateur pourrait saisir séparés par un espace, et observe celui vers lequel le navigateur défile, il peut ajouter ce caractère au terme de recherche et mettre à jour la zone de défilement pour construire progressivement le terme de recherche. En enveloppant chaque prochaine correspondance possible dans un [details](#)élément fermé, la page pourrait écouter [toggle](#)les événements au lieu de [scroll](#) événements. Cette attaque pourrait être traitée pour les deux événements en n'agissant pas sur chaque caractère saisi par l'utilisateur dans la boîte de dialogue de recherche dans la page.*

### 6.9.3 Interaction avec la sélection

Le processus de recherche dans la page est invoqué dans le contexte d'un document et peut avoir un effet sur la [sélection](#) de ce document. Plus précisément, la plage qui définit la [correspondance active](#) peut dicter la sélection actuelle. Ces mises à jour de sélection, cependant, peuvent se produire à différents moments au cours du processus de recherche dans la page (par exemple lors du rejet [de l'interface de recherche dans la page](#) ou lors d'un changement dans la plage [de correspondance active](#) ).

## 6.10 Glisser-déposer



Cette section définit un mécanisme de glisser-déposer basé sur les événements.

Cette spécification ne définit pas exactement ce qu'est réellement une *opération de glisser-déposer* .

Sur un support visuel doté d'un dispositif de pointage, une opération de glisser pourrait être l'action par défaut d'un [mousedown](#)événement suivi d'une série

d' `mousemove` événements, et le déplacement pourrait être déclenché par le relâchement de la souris.

Lors de l'utilisation d'une modalité d'entrée autre qu'un dispositif de pointage, les utilisateurs devraient probablement indiquer explicitement leur intention d'effectuer une opération de glisser-déposer, en indiquant respectivement ce qu'ils souhaitent faire glisser et où ils souhaitent le déposer.

Quelle que soit sa mise en œuvre, les opérations de glisser-déposer doivent avoir un point de départ (par exemple, l'endroit où la souris a été cliquée, ou le début de la sélection ou de l'élément qui a été sélectionné pour le glisser), peuvent avoir un nombre quelconque d'étapes intermédiaires (éléments que la souris se déplace pendant un glissement, ou des éléments que l'utilisateur sélectionne comme points de chute possibles au fur et à mesure qu'il parcourt les possibilités), et doit avoir un point final (l'élément au-dessus duquel le bouton de la souris a été relâché, ou l'élément qui a finalement été sélectionné), ou être annulé. Le point final doit être le dernier élément sélectionné comme point de chute possible avant que la chute ne se produise (donc si l'opération n'est pas annulée, il doit y avoir au moins un élément dans l'étape intermédiaire).

### 6.10.1 Présentation

*Cette section est non normative.*

Pour rendre un élément déplaçable, donnez à l'élément un `draggable` attribut et définissez un écouteur d'événement pour `dragstart` qui stocke les données glissées.

Le gestionnaire d'événements doit généralement vérifier qu'il ne s'agit pas d'une sélection de texte qui est glissée, puis doit stocker des données dans l' `DataTransfer` objet et définir les effets autorisés (copie, déplacement, lien ou une combinaison).

Par exemple:

```
<p>What fruits do you like?</p>
<ol ondragstart="dragStartHandler(event)">
  <li draggable="true" data-value="fruit-apple">Apples</li>
  <li draggable="true" data-value="fruit-orange">Oranges</li>
  <li draggable="true" data-value="fruit-pear">Pears</li>
</ol>
<script>
  var internalDNDType = 'text/x-example'; // set this to something
  specific to your site

  function dragStartHandler(event) {
    if (event.target instanceof HTMLLIElement) {
```

```

        // use the element's data-value="" attribute as the value to
        be moving:

        event.dataTransfer.setData(internalDNDType,
        event.target.dataset.value);

        event.dataTransfer.effectAllowed = 'move'; // only allow
        moves

    } else {

        event.preventDefault(); // don't allow selection to be
        dragged

    }

}

</script>

```

---

Pour accepter un dépôt, la cible de dépôt doit écouter les événements suivants :

1. Le [dragenter](#) gestionnaire d'événements signale si la cible de dépôt est potentiellement disposée à accepter le dépôt, en annulant l'événement.
2. Le [dragover](#) gestionnaire d'événements spécifie les commentaires qui seront affichés à l'utilisateur, en définissant l' [dropEffect](#) attribut [DataTransfer](#) associé à l'événement. Cet événement doit également être annulé.
3. Le [drop](#) gestionnaire d'événements a une dernière chance d'accepter ou de rejeter la suppression. Si la suppression est acceptée, le gestionnaire d'événements doit effectuer l'opération de suppression sur la cible. Cet événement doit être annulé, afin que la [dropEffect](#) valeur de l'attribut puisse être utilisée par la source. Sinon, l'opération de suppression est rejetée.

Par exemple:

```

<p>Drop your favorite fruits below:</p>
<ol ondragenter="dragEnterHandler(event)"
ondragover="dragOverHandler(event)"
    ondrop="dropHandler(event)">
</ol>
<script>
    var internalDNDType = 'text/x-example'; // set this to something
    specific to your site

    function dragEnterHandler(event) {
        var items = event.dataTransfer.items;
        for (var i = 0; i < items.length; ++i) {
            var item = items[i];

```

```

        if (item.kind == 'string' && item.type == internalDNDType) {
            event.preventDefault();
            return;
        }
    }

    function dragOverHandler(event) {
        event.dataTransfer.dropEffect = 'move';
        event.preventDefault();
    }

    function dropHandler(event) {
        var li = document.createElement('li');
        var data = event.dataTransfer.getData(internalDNDType);
        if (data == 'fruit-apple') {
            li.textContent = 'Apples';
        } else if (data == 'fruit-orange') {
            li.textContent = 'Oranges';
        } else if (data == 'fruit-pear') {
            li.textContent = 'Pears';
        } else {
            li.textContent = 'Unknown Fruit';
        }
        event.target.appendChild(li);
    }
</script>

```

---

Pour supprimer l'élément d'origine (celui qui a été glissé) de l'affichage, l' `dragend` événement peut être utilisé.

Pour notre exemple ici, cela signifie mettre à jour le balisage d'origine pour gérer cet événement :

```

<p>What fruits do you like?</p>

<ol ondragstart="dragStartHandler(event)"
ondragend="dragEndHandler(event)">

    ...as before...

</ol>

<script>

```

```
function dragStartHandler(event) {
    // ...as before...
}

function dragEndHandler(event) {
    if (event.dataTransfer.dropEffect == 'move') {
        // remove the dragged element
        event.target.parentNode.removeChild(event.target);
    }
}
</script>
```

### 6.10.2 Le magasin de données de glissement

Les données sous-jacentes à une opération de glisser-déposer, appelées **magasin de données de glissement**, se composent des informations suivantes :

- Une **liste d'éléments de magasin de données de glissement**, qui est une liste d'éléments représentant les données glissées, chacune comprenant les informations suivantes :

#### ***Le type d'élément de données glisser***

Le type de données :

##### ***Texte***

Texte.

##### ***Déposer***

Données binaires avec un nom de fichier.

#### ***La chaîne de type d'élément de données glisser***

Une chaîne Unicode donnant le type ou le format des données, généralement donné par un [type MIME](#). Certaines valeurs qui ne sont pas [des types MIME](#) ont une casse spéciale pour des raisons héritées. L'API n'impose pas l'utilisation des [types MIME](#) ; d'autres valeurs peuvent également être utilisées. Dans tous les cas, cependant, les valeurs sont toutes [converties en minuscules ASCII](#) par l'API.

Il existe une limite d'un élément *de texte* par [chaîne de type d'élément](#).

#### **Les données réelles**

Une chaîne Unicode ou binaire, dans certains cas avec un nom de fichier (lui-même une chaîne Unicode), selon [le type d'élément de données glisser](#).



La [liste des éléments du magasin de données de glissement](#) est triée dans l'ordre dans lequel les éléments ont été ajoutés à la liste ; le plus récemment ajouté en dernier.

- Les informations suivantes, utilisées pour générer les commentaires de l'interface utilisateur pendant le glissement :
  - Informations de retour par défaut définies par l'agent utilisateur, appelées **retours par défaut du magasin de données de glissement** .
  - Facultativement, une image bitmap et la coordonnée d'un point dans cette image, appelée **image bitmap du magasin de données de glissement** et **coordonnée du point chaud du magasin de données de glissement** .
- Un **mode de magasin de données glisser** , qui est l'un des suivants :

#### ***Mode lecture/écriture***

Pour l' [dragstart](#) événement. De nouvelles données peuvent être ajoutées au [magasin de données de glissement](#) .

#### ***Mode lecture seule***

Pour l' [drop](#) événement. La liste des éléments représentant les données glissées peut être lue, y compris les données. Aucune nouvelle donnée ne peut être ajoutée.

#### ***Mode protégé***

Pour tous les autres événements. Les formats et les types dans la liste des éléments [du magasin de données de glissement](#) représentant les données glissées peuvent être énumérés, mais les données elles-mêmes ne sont pas disponibles et aucune nouvelle donnée ne peut être ajoutée.

- Un **magasin de données de glissement a autorisé les effets state** , qui est une chaîne.

Lorsqu'un [magasin de données de glissement](#) est **créé** , il doit être initialisé de sorte que sa [liste d'éléments de magasin de données de glissement](#) soit vide, qu'il n'ait pas [de retour par défaut du magasin de données de glissement](#) , qu'il n'ait pas [de bitmap de magasin de données de glissement](#) et [de coordonnée de point chaud du magasin de données de glissement](#) , ses [données de glissement le mode de stockage](#) est [le mode protégé](#) et son [état d'effets autorisés pour le stockage de données de glissement](#) est la chaîne " [uninitialized](#)".

### 6.10.3 L' [DataTransfer](#) interface



DataTransfer les objets sont utilisés pour exposer le magasin de données de glissement qui sous-tend une opération de glisser-déposer.

```
[Exposed=Window]
```

```
interface DataTransfer {
```

```
    constructor();
```

```
    attribute DOMString dropEffect;
```

```
    attribute DOMString effectAllowed;
```

```
    [SameObject] readonly attribute DataTransferItemList items;
```

```
    undefined setDragImage(Element image, long x, long y);
```

```
    /* old interface */
```

```
    readonly attribute FrozenArray<DOMString> types;
```

```
    DOMString getData(DOMString format);
```

```
    undefined setData(DOMString format, DOMString data);
```

```
    undefined clearData(optional DOMString format);
```

```
    [SameObject] readonly attribute FileList files;
```

```
};
```

```
dataTransfer = new DataTransfer()
```

✓

Crée un nouvel DataTransfer objet avec un magasin de données de glissement vide .

```
dataTransfer.dropEffect [ = value ]
```

✓

Renvoie le type d'opération actuellement sélectionné. Si le type d'opération ne fait pas partie de ceux autorisés par l' `effectAllowed` attribut, l'opération échouera.

Peut être défini pour modifier l'opération sélectionnée.

Les valeurs possibles sont "`none`", "`copy`", "`link`" et "`move`".

`dataTransfer.effectAllowed` [ = *value* ]

✓ 

Renvoie les types d'opérations qui doivent être autorisées.

Peut être défini (pendant l' `dragstart` événement) pour modifier les opérations autorisées.

Les valeurs possibles sont "`none`", "`copy`", "`copyLink`", "`copyMove`", "`link`", "`linkMove`", "`move`", "`all`", et "`uninitialized`",

`dataTransfer.items`

✓ 

Renvoie un `DataTransferItemList` objet, avec les données de glissement.

`dataTransfer.setDragImage` (*element*, *x*, *y*)

✓ 

Utilise l'élément donné pour mettre à jour le retour de glissement, en remplaçant tout retour précédemment spécifié.

`dataTransfer.types`

✓ 

Renvoie un [tableau figé](#) répertoriant les formats qui ont été définis dans l' `dragstart` événement. De plus, si des fichiers sont déplacés, l'un des types sera la chaîne "`Files`".

`data = dataTransfer.getData` (*format*)

✓ 

Renvoie les données spécifiées. S'il n'y a pas de telles données, renvoie la chaîne vide.

`dataTransfer.setData` (*format*, *data*)

✓ 

Ajoute les données spécifiées.

`dataTransfer.clearData` ([ *format* ])

✓ 

Supprime les données des formats spécifiés. Supprime toutes les données si l'argument est omis.

`dataTransfer.files`



Renvoie un `FileList` des fichiers déplacés, le cas échéant.

`DataTransfer` les objets créés dans le cadre d' [événements de glisser-déposer](#) ne sont valides que pendant le déclenchement de ces événements.

Un `DataTransfer` objet est associé à un [magasin de données de glissement](#) tant qu'il est valide.

Un `DataTransfer` objet a un **tableau de types** associé, qui est un `FrozenArray<DOMString>`, initialement vide. Lorsque le contenu de la [liste des éléments du magasin de données de glissement](#) `DataTransfer` de l'objet change ou lorsque l' objet n'est plus associé à un [magasin de données de glissement](#), exécutez les étapes suivantes : `DataTransfer`

1. Soit *L* une suite vide.
2. Si l' `DataTransfer` objet est toujours associé à un [magasin de données de glissement](#), alors :
  1. Pour chaque élément de la [liste d'éléments du magasin de données de glissement](#) `DataTransfer` de l'objet dont [le type](#) est *text*, ajoutez une entrée à *L* constituée du [type string](#) de l'élément.
  2. S'il existe des éléments dans la [liste d'éléments du magasin de données de glissement](#) `DataTransfer` de l'objet dont [le type](#) est *File*, ajoutez une entrée à *L* consistant en la chaîne " ". (Cette valeur peut être distinguée des autres valeurs car elle n'est pas en minuscule.) `Files`
3. Définissez le [tableau des types](#) `DataTransfer` de l'objet sur le résultat de [la création d'un tableau gelé](#) à partir de *L*.

Le `DataTransfer()` constructeur, lorsqu'il est appelé, doit renvoyer un `DataTransfer` objet nouvellement créé initialisé comme suit :

1. Définissez la [liste d'éléments](#) du [magasin de données de glissement](#) comme une liste vide.
2. Définissez le [mode](#) du [magasin de données de glissement](#) sur le [mode lecture/écriture](#).
3. Réglez le `dropEffect` et `effectAllowed` sur "aucun".

L' `dropEffect` attribut contrôle les commentaires de glisser-déposer que l'utilisateur reçoit lors d'une opération de glisser-déposer. Lorsque l' `DataTransfer` objet est créé, l' `dropEffect` attribut est défini sur une valeur de chaîne. Lors de l'obtention, il doit retourner sa valeur actuelle. Lors du paramétrage, si la nouvelle valeur est " " **none**,

" **copy**", " **link**" ou " **move**", la valeur actuelle de l'attribut doit être définie sur la nouvelle valeur. Les autres valeurs doivent être ignorées.

L' **effectAllowed** attribut est utilisé dans le modèle de traitement par glisser-déposer pour initialiser l' **dropEffect** attribut pendant les événements **dragenter** et **dragover**. Lorsque l' **DataTransfer** objet est créé, l' **effectAllowed** attribut est défini sur une valeur de chaîne. Lors de l'obtention, il doit retourner sa valeur actuelle. Lors du réglage, si le **mode** de **magasin de données par glissement** est le **mode de lecture/écriture** et que la nouvelle valeur est l'une des valeurs " ", " ", " ", " ", " ", " ", " ", " ", " " ou " ", la valeur actuelle de l'attribut doit être définie sur la nouvelle valeur. Sinon, il doit rester inchangé. **none copy copyLink copyMove link linkMove move all uninitialized**

L' **items** attribut doit renvoyer un **DataTransferItemList** objet associé à l' **DataTransfer** objet.

La méthode doit exécuter les étapes suivantes : **setDragImage (image, x, y)**

1. Si l' **DataTransfer** objet n'est plus associé à un **magasin de données de glissement** , revenez. Il ne se passe rien.
2. Si le **mode** du **magasin de données de glissement** n'est pas le **mode lecture/écriture** , retournez. Il ne se passe rien.
3. Si *image* est un **img** élément, définissez le **bitmap du magasin de données de glissement** sur l'image de l'élément (à sa **taille intrinsèque** ) ; sinon, définissez le **bitmap du magasin de données de glissement** sur une image générée à partir de l'élément donné (le mécanisme exact pour le faire n'est pas actuellement spécifié).
4. Définissez la **coordonnée du point chaud du magasin de données de glissement** sur la coordonnée *x* , *y* donnée .

L' **types** attribut doit renvoyer le **tableau des types** **DataTransfer** de cet objet .

La méthode doit exécuter les étapes suivantes : **getData (format)**

1. Si l' **DataTransfer** objet n'est plus associé à un **magasin de données de glissement** , renvoyez la chaîne vide.
2. Si le **mode** du **magasin de données de glissement** est le **mode protégé** , renvoyez la chaîne vide.
3. Soit *format* le premier argument, **converti en ASCII minuscule** .
4. Laissez *convert-to-URL* être faux.
5. Si le *format* est égal à " *text*", remplacez-le par " *text/plain*".

6. Si *format* est égal à " `url`", remplacez-le par " `text/uri-list`" et définissez *convert-to-URL* sur `true`.
7. S'il n'y a aucun élément dans la [liste d'éléments du magasin de données de glissement](#) dont [le type](#) est *text* et dont [la chaîne de type](#) est égale à *format*, renvoie la chaîne vide.
8. Soit *result* les données de l'élément dans la [liste d'éléments du magasin de données de glissement](#) dont [le type](#) est une chaîne Unicode simple et dont [la chaîne de type](#) est égale à *format*.
9. Si *convert-to-URL* est vrai, analysez *le résultat* en fonction des `text/uri-list` données, puis définissez *le résultat* sur la première URL de la liste, le cas échéant, ou sur la chaîne vide dans le cas contraire. [\[RFC2483\]](#)
10. Retourner *le résultat*.

La méthode doit exécuter les étapes suivantes : *setData (format, data)*

1. Si l' `DataTransfer` objet n'est plus associé à un [magasin de données de glissement](#), revenez. Il ne se passe rien.
2. Si le [mode](#) du [magasin de données de glissement](#) n'est pas le [mode lecture/écriture](#), retournez. Il ne se passe rien.
3. Soit *format* le premier argument, [converti en ASCII minuscule](#).
4. Si *le format* est égal à " `text`", remplacez-le par " `text/plain`".  
Si *le format* est égal à " `url`", remplacez-le par " `text/uri-list`".
5. Supprimez l'élément de [la liste des éléments du magasin de données de glissement](#) dont [le type](#) est *text* et dont [la chaîne de type](#) est égale à *format*, s'il en existe un.
6. Ajoutez un élément à la [liste des éléments du magasin de données de glissement](#) dont [le type](#) est *text*, dont [la chaîne de type](#) est égale à *format* et dont les données sont la chaîne donnée par le deuxième argument de la méthode.

La méthode doit exécuter les étapes suivantes : *clearData (format)*

1. Si l' `DataTransfer` objet n'est plus associé à un [magasin de données de glissement](#), revenez. Il ne se passe rien.
2. Si le [mode](#) du [magasin de données de glissement](#) n'est pas le [mode lecture/écriture](#), retournez. Il ne se passe rien.
3. Si la méthode a été appelée sans arguments, supprimez chaque élément de la [liste d'éléments du magasin de données de glissement](#) dont [le type](#) est *Plain Unicode string*, et retournez.

4. Définissez *format* sur *format* , [converti en minuscules ASCII](#) .
5. Si le *format* est égal à " `text`", remplacez-le par " `text/plain`".  
Si le *format* est égal à " `url`", remplacez-le par " `text/uri-list`".
6. Supprimez l'élément de [la liste des éléments du magasin de données de glissement](#) dont [le type](#) est *text* et dont [la chaîne de type](#) est égale à *format* , s'il en existe un.

*La `clearData()` méthode n'affecte pas si des fichiers ont été inclus dans le glissement, de sorte que la `types` liste de l'attribut peut toujours ne pas être vide après l'appel `clearData()` (elle contiendrait toujours la `Files` chaîne " " si des fichiers étaient inclus dans le glissement).*

L' **files** attribut doit renvoyer une séquence [en direct](#) composée d' objets représentant les fichiers trouvés par les étapes suivantes. De plus, pour un objet donné et un fichier sous-jacent donné, le même objet doit être utilisé à chaque fois. `FileListFileFileListFile`

1. Commencez avec une liste vide *L* .
2. Si l' `DataTransfer` objet n'est plus associé à un [magasin de données de glissement](#) , le `FileList` est vide. Renvoie la liste vide *L* .
3. Si le [mode](#) du [magasin de données de glissement](#) est le [mode protégé](#) , renvoie la liste vide *L* .
4. Pour chaque élément de la [liste d'éléments du magasin de données de glissement](#) dont [le type](#) est *File* , ajoutez les données de l'élément (le fichier, en particulier son nom et son contenu, ainsi que son [type](#) ) à la liste *L* .
5. Les fichiers trouvés par ces étapes sont ceux de la liste *L* .

*Cette version de l'API n'expose pas les types de fichiers lors du glisser.*

#### 6.10.3.1 L' `DataTransferItemList` interface



Chaque `DataTransfer` objet est associé à un `DataTransferItemList` objet.

```
[Exposed=Window]
```

```
interface DataTransferItemList {
```

```
readonly attribute unsigned long length;
```

```
getter DataTransferItem (unsigned long index);
```

```
DataTransferItem? add(DOMString data, DOMString type);
```

```
DataTransferItem? add(File data);
```

```
undefined remove(unsigned long index);
```

```
undefined clear();
```

```
};
```

***items . [length](#)***

✓ 

Renvoie le nombre d'éléments dans le [magasin de données de glissement](#) .

***items [index]***

Renvoie l' [DataTransferItem](#) objet représentant la ième entrée d'index dans le [magasin de données de glissement](#) .

***items . [remove](#) (index)***

✓ 

Supprime la ième entrée d'index dans le [magasin de données de glissement](#) .

***items . [clear](#) ()***

✓ 

Supprime toutes les entrées du [magasin de données de glissement](#) .

***items . [add](#) (data)***

✓ 

***items . [add](#) (data , type)***

Ajoute une nouvelle entrée pour les données données au [magasin de données de glissement](#) . Si les données sont en texte brut, une chaîne de type doit également être fournie.

Alors que l' objet [DataTransferItemList](#) de l'objet [DataTransfer](#) est associé à un [magasin de données de glissement](#) , le *mode* [DataTransferItemList](#) de l'objet est le même que le [mode de magasin de données de glissement](#) . Lorsque l' objet de l'objet n'est pas associé à un [magasin de données de glissement](#) , le *mode* de l'objet est le *mode désactivé* . Le [magasin de données de glissement](#) référencé dans cette section (qui est utilisé uniquement lorsque l' objet n'est pas en *mode désactivé* ) est le [magasin de données de glissement](#) auquel l' objet de l'objet est associé. [DataTransferItemListDataTransferDataTransferItemListDataTransferItem](#)  
[ListDataTransferItemListDataTransfer](#)



L' **length** attribut doit retourner zéro si l'objet est en *mode désactivé* ; sinon, il doit renvoyer le nombre d'éléments dans la [liste d'éléments du magasin de données de glissement](#) .

Lorsqu'un [DataTransferItemList](#) objet n'est pas en *mode désactivé* , ses [indices de propriété pris en charge](#) sont les [indices](#) de la [liste d'éléments du magasin de données de glissement](#) .

Pour [déterminer la valeur d'une propriété indexée](#) *i* d'un [DataTransferItemList](#) objet, l'agent utilisateur doit retourner un [DataTransferItem](#) objet représentant le *i* ème élément dans le [magasin de données de glissement](#) . Le même objet doit être renvoyé chaque fois qu'un élément particulier est obtenu à partir de cet [DataTransferItemList](#) objet. L' [DataTransferItem](#) objet doit être associé au même [DataTransfer](#) objet que l' [DataTransferItemList](#) objet lors de sa première création.

La **add ()** méthode doit exécuter les étapes suivantes :

1. Si l' [DataTransferItemList](#) objet n'est pas en [mode lecture/écriture](#) , renvoie null.
2. Passez à l'ensemble d'étapes approprié dans la liste suivante :

**Si le premier argument de la méthode est une chaîne**

S'il existe déjà un élément dans la [liste d'éléments du magasin de données de glissement](#) dont [le type](#) est *du texte* et dont [la chaîne de type](#) est égale à la valeur du deuxième argument de la méthode, [converti en ASCII minuscule](#) , lancez un ["NotSupportedError"](#) [DOMException](#) .

Sinon, ajoutez un élément à la [liste des éléments du magasin de données de glissement](#) dont [le type](#) est *text* , dont [la chaîne de type](#) est égale à la valeur du deuxième argument de la méthode, [convertie en ASCII minuscule](#) et dont les données sont la chaîne donnée par le premier argument de la méthode.

**Si le premier argument de la méthode est un [File](#)**

Ajoutez un élément à la [liste des éléments du magasin de données de glissement](#) dont [le type](#) est *File* , dont [la chaîne de type](#) est le [type](#) de [File](#), [converti en ASCII minuscule](#) et dont les données sont identiques à celles [File](#) de .

3. [Déterminez la valeur de la propriété indexée](#) correspondant à l'élément nouvellement ajouté et renvoyez cette valeur (un [DataTransferItem](#) objet nouvellement créé).

La méthode doit exécuter ces étapes : **remove (*index*)**

1. Si l' [DataTransferItemList](#) objet n'est pas en [mode lecture/écriture](#) , lancer un ["InvalidStateError"](#) [DOMException](#) .

2. Si le [magasin de données de glissement](#) ne contient pas d'élément *d'index* ème, alors retournez.
3. Supprimez l' *index* e élément du [magasin de données de glissement](#) .

La **clear()** méthode, si l' [DataTransferItemList](#) objet est en [mode lecture/écriture](#) , doit supprimer tous les éléments du [magasin de données de glissement](#) . Sinon, il ne doit rien faire.

### 6.10.3.2 L' [DataTransferItem](#) interface



Chaque [DataTransferItem](#) objet est associé à un [DataTransfer](#) objet.

```
[Exposed=Window]

interface DataTransferItem {

    readonly attribute DOMString kind;

    readonly attribute DOMString type;

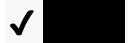
    undefined getAsString(FunctionStringCallback? _callback);

    File? getAsFile();

};

callback FunctionStringCallback = undefined (DOMString data);
```

***item.kind***



Renvoie [le type de l'élément de données de glissement](#) , l'un des suivants : "chaîne", "fichier".

***item.type***



Renvoie [la chaîne de type d'élément de données glisser](#) .

***item.getAsString(callback)***



Appelle le rappel avec les données de chaîne comme argument, si [le type de l'élément de données de glissement](#) est *text* .

```
file = item.getAsFile()
```

✓

Renvoie un [File](#)objet, si [le type de l'élément de données de glissement](#) est *File* .

Bien que l' objet [DataTransferItem](#) de l'objet [DataTransfer](#) soit associé à un [magasin de données de glissement](#) et que [la liste d'éléments du magasin de données de glissement du magasin de données de glissement](#) contienne [toujours](#) l'élément représenté par l'objet, le *mode* de l'objet est le même que le [mode de magasin de données de glissement](#) . Lorsque l' objet de l'objet *n'est pas* associé à un [magasin de données de glissement](#) , ou si l'élément représenté par l'objet a été supprimé de la [liste d'éléments de magasin de données de glissement](#) pertinente , le *mode* de l'objet est le *mode désactivé* . Le [DataTransferItemDataTransferItemDataTransferItemDataTransferDataTransferItemDataTransferItem](#) [le magasin de données de glissement](#) référencé dans cette section (qui est utilisé uniquement lorsque l' [DataTransferItem](#) objet n'est pas en *mode désactivé* ) est le [magasin de données de glissement](#) auquel l' objet [DataTransferItem](#) de l'objet [DataTransfer](#) est associé.

L' **kind** attribut doit retourner la chaîne vide si l' [DataTransferItem](#) objet est en *mode désactivé* ; sinon, il doit renvoyer la chaîne indiquée dans la cellule de la deuxième colonne du tableau suivant à partir de la ligne dont la cellule de la première colonne contient [le type d'élément de données glisser](#) de l'élément représenté par l' [DataTransferItem](#) objet :

Type	Chaîne
Texte	" string "
Déposer	" file "

L' **type** attribut doit retourner la chaîne vide si l' [DataTransferItem](#) objet est en *mode désactivé* ; sinon, il doit renvoyer [la chaîne de type d'élément de données glisser](#) de l'élément représenté par l' [DataTransferItem](#) objet.

La méthode doit exécuter les étapes suivantes : [getAsString \(callback\)](#)

1. Si le *rappel* est nul, retour.
2. Si l' [DataTransferItem](#) objet n'est pas en [mode lecture/écriture](#) ou en [mode lecture seule](#) , retour. Le rappel n'est jamais invoqué.
3. Si [le type d'élément de données de glissement](#) n'est pas *text* , alors retournez. Le rappel n'est jamais invoqué.

4. Sinon, [mettez une tâche en file d'attente](#) pour appeler *callback* , en transmettant les données réelles de l'élément représenté par l' [DataTransferItem](#) objet comme argument.

La [getAsFile\(\)](#) méthode doit exécuter les étapes suivantes :

1. Si l' [DataTransferItem](#) objet n'est pas en [mode lecture/écriture](#) ou en [mode lecture seule](#) , alors renvoie null.
2. Si [le type d'élément de données de glissement](#) n'est pas *File* , renvoie null.
3. Renvoie un nouvel [File](#) objet représentant les données réelles de l'élément représenté par l' [DataTransferItem](#) objet.

#### 6.10.4 L' [DragEvent](#) interface



Le modèle de traitement par glisser-déposer implique plusieurs événements. Ils utilisent tous l' [DragEvent](#) interface.

```
[Exposed=Window]
```

```
interface DragEvent : MouseEvent {
```

```
  constructor(DOMString type, optional DragEventInit
```

```
  eventInitDict = {});
```

```
  readonly attribute DataTransfer? dataTransfer;
```

```
};
```

```
dictionary DragEventInit : MouseEventInit {
```

```
  DataTransfer? dataTransfer = null;
```

```
};
```

***event.dataTransfer***



Renvoie l' [DataTransfer](#) objet de l'événement.

Bien que, pour des raisons de cohérence avec d'autres interfaces d'événement, l' [DragEvent](#) interface ait un constructeur, il n'est pas particulièrement utile. En particulier, il n'y a aucun moyen de créer un [DataTransfer](#) objet utile à partir d'un script, car [DataTransfer](#) les objets ont un modèle de traitement et de sécurité coordonné par le navigateur lors des glisser-déposer.

L' [dataTransfer](#) attribut de l' [DragEvent](#) interface doit renvoyer la valeur à laquelle il a été initialisé. Il représente les informations contextuelles de l'événement.

Lorsqu'un agent utilisateur doit **déclencher un événement NPD** nommé *e* sur un élément, à l'aide d'un [magasin de données de glissement](#) particulier et éventuellement avec une *cible* associée spécifique, l'agent utilisateur doit exécuter les étapes suivantes :

1. Laissez *dataDragStoreWasChanged* être faux.
2. Si aucune *cible* associée spécifique n'a été fournie, définissez *la cible associée* sur null.
3. Soit *window* l' [objet global pertinent](#) de l' [Document](#) objet de l'élément cible spécifié.
4. Si *e* vaut [dragstart](#), définissez le [mode de stockage des données de glissement](#) sur le [mode lecture/écriture](#) et définissez *dataDragStoreWasChanged* sur true.

Si *e* vaut [drop](#), définissez le [mode de stockage des données par glissement](#) sur le [mode lecture seule](#) .

5. Soit *dataTransfer* un objet nouvellement créé [DataTransfer](#) associé au [magasin de données de glissement](#) donné .
6. Définissez l' [effectAllowed](#) attribut sur [l'état des effets autorisés du magasin de données de glissement](#) du magasin de données de glissement .
7. Définissez l' [dropEffect](#) attribut sur "[none](#)" si *e* est [dragstart](#), [drag](#), ou [dragleave](#); à la valeur correspondant à l' [opération de glissement en cours](#) si *e* vaut [drop](#) ou [dragend](#); et à une valeur basée sur la [effectAllowed](#) valeur de l'attribut et la source du glisser-déposer, comme indiqué par le tableau suivant, sinon (c'est-à-dire si *e* est [dragenter](#) ou [dragover](#)) :

<a href="#">effectAllowed</a>	<a href="#">dropEffect</a>
" <a href="#">none</a> "	" <a href="#">none</a> "
" <a href="#">copy</a> "	" <a href="#">copy</a> "
" <a href="#">copyLink</a> "	" <a href="#">copy</a> ", ou, <a href="#">le cas échéant</a> , " <a href="#">link</a> "
" <a href="#">copyMove</a> "	" <a href="#">copy</a> ", ou, <a href="#">le cas échéant</a> , " <a href="#">move</a> "

<u>effectAllowed</u>	<u>dropEffect</u>
" <u>all</u> "	" <u>copy</u> ", ou, <u>le cas échéant</u> , soit " <u>link</u> " ou " <u>move</u> "
" <u>link</u> "	" <u>link</u> "
" <u>linkMove</u> "	" <u>link</u> ", ou, <u>le cas échéant</u> , " <u>move</u> "
" <u>move</u> "	" <u>move</u> "
" <u>uninitialized</u> " lorsque ce qui est déplacé est une sélection d'un contrôle de texte	" <u>move</u> ", ou, <u>le cas échéant</u> , soit " <u>copy</u> " ou " <u>link</u> "
" <u>uninitialized</u> " lorsque ce qui est déplacé est une sélection	" <u>copy</u> ", ou, <u>le cas échéant</u> , soit " <u>link</u> " ou " <u>move</u> "
" <u>uninitialized</u> " lorsque ce qui est déplacé est un <u>a</u> élément avec un <u>href</u> attribut	" <u>link</u> ", ou, <u>le cas échéant</u> , soit " <u>copy</u> " ou " <u>move</u> "
Tout autre cas	" <u>copy</u> ", ou, <u>le cas échéant</u> , soit " <u>link</u> " ou " <u>move</u> "

8. Lorsque le tableau ci-dessus fournit **des alternatives éventuellement appropriées**, les agents utilisateurs peuvent utiliser à la place les valeurs alternatives répertoriées si les conventions de la plate-forme dictent que l'utilisateur a demandé ces effets alternatifs.
9. Par exemple, les conventions de la plate-forme Windows sont telles que faire glisser tout en maintenant la touche "alt" enfoncée indique une préférence pour lier les données, plutôt que de les déplacer ou de les copier. Par conséquent, sur un système Windows, si "link" est une option selon le tableau ci-dessus alors que la touche "alt" est enfoncée, l'agent utilisateur peut la sélectionner au lieu de "copy" ou "move".
10. Soit *événement* le résultat de [la création d'un événement](#) à l'aide de DragEvent.
11. Initialisez l'attribut eventType à *e*, son bubblesattribut à *true*, son viewattribut à *window*, son relatedTargetattribut à *related target* et son dataTransferattribut à *dataTransfer*.
12. Si *e* n'est pas dragleave ou dragend, alors initialisez l'attribut de l'événement cancelable à *true*.
13. Initialiser les attributs souris et clé de l'événement initialisés en fonction de l'état des périphériques d'entrée comme ils le seraient pour les événements d'interaction utilisateur.  
  
S'il n'y a pas de périphérique de pointage pertinent, initialisez les attributs `screenX`, `screenY`, `clientX` et `clientY` à 0.
14. [Envoyer](#) l'événement à l'élément cible spécifié.

15. Définissez l' [état des effets autorisés du magasin de données de glissement](#) sur la valeur actuelle de l' attribut `dataTransfer`[effectAllowed](#) . (Il ne peut avoir changé de valeur que si e vaut [dragstart](#).)
16. Si `dataDragStoreWasChanged` a la valeur true, redéfinissez le [mode de stockage des données de glissement](#) sur le [mode protégé](#) .
17. Rompre l'association entre `dataTransfer` et le [magasin de données glisser](#) .

### 6.10.5 Modèle de traitement

Lorsque l'utilisateur tente de commencer une opération de glissement, l'agent utilisateur doit exécuter les étapes suivantes. Les agents utilisateurs doivent agir comme si ces étapes étaient exécutées même si le glissement a réellement commencé dans un autre document ou une autre application et que l'agent utilisateur n'était pas conscient que le glissement se produisait jusqu'à ce qu'il croise un document sous la responsabilité de l'agent utilisateur.

1. Déterminez ce qui est déplacé, comme suit :

Si l'opération de glisser a été invoquée sur une sélection, c'est la sélection qui est glissée.

Sinon, si l'opération de glissement a été invoquée sur un [Document](#), c'est le premier élément, remontant la chaîne d'ancêtres, commençant au nœud que l'utilisateur a essayé de faire glisser, dont l'attribut IDL est [draggable](#) défini sur true. S'il n'y a pas un tel élément, alors rien n'est traîné; return, l'opération de glisser-déposer n'est jamais démarrée.

Sinon, l'opération de glissement a été invoquée en dehors de la compétence de l'agent utilisateur. Ce qui est déplacé est défini par le document ou l'application où le glissement a été lancé.

*[img](#) les éléments et [a](#) les éléments avec un [href](#) attribut ont leur [draggable](#) attribut défini sur true par défaut.*

2. [Créez un magasin de données de glissement](#) . Tous les événements NPD déclenchés ultérieurement par les étapes de cette section doivent utiliser ce [magasin de données de glissement](#) .
3. Établissez quel nœud DOM est le **nœud source** , comme suit :

S'il s'agit d'une sélection qui est glissée, le [nœud source](#) est le [Text](#) nœud sur lequel l'utilisateur a commencé le glissement (généralement le [Text](#) nœud sur lequel l'utilisateur a initialement cliqué). Si l'utilisateur n'a pas spécifié de nœud particulier, par exemple s'il a simplement dit à l'agent utilisateur de commencer à faire glisser "la sélection", alors le [nœud source](#) est le premier [Text](#) nœud contenant une partie de la sélection.

Sinon, si c'est un élément qui est glissé, alors le [nœud source](#) est l'élément qui est glissé.

Sinon, le [nœud source](#) fait partie d'un autre document ou d'une autre application. Lorsque cette spécification exige qu'un événement soit distribué au [nœud source](#) dans ce cas, l'agent utilisateur doit à la place suivre les conventions spécifiques à la plate-forme pertinentes pour cette situation.

*Plusieurs événements sont déclenchés sur le [nœud source](#) au cours de l'opération de glisser-déposer.*

4. Déterminez la **liste des nœuds déplacés** , comme suit :

S'il s'agit d'une sélection qui est glissée, alors la [liste des nœuds glissés](#) contient, dans [l'ordre de l'arborescence](#) , tous les nœuds partiellement ou complètement inclus dans la sélection (y compris tous leurs ancêtres).

Sinon, la [liste des nœuds glissés](#) ne contient que le [nœud source](#) , le cas échéant.

5. S'il s'agit d'une sélection qui est déplacée, ajoutez un élément à la [liste d'éléments du magasin de données de glissement](#) , avec ses propriétés définies comme suit :

#### La chaîne de type d'élément de données glisser

" [text/plain](#) "

#### Le type d'élément de données glisser

Texte

#### **Les données réelles**

Le texte de la sélection

Sinon, si des fichiers sont déplacés, ajoutez un élément par fichier à la [liste des éléments du magasin de données de glissement](#) , avec leurs propriétés définies comme suit :

#### La chaîne de type d'élément de données glisser

Le type MIME du fichier, s'il est connu, ou " [application/octet-stream](#) " sinon.

#### Le type d'élément de données glisser

Déposer

#### **Les données réelles**

Le contenu et le nom du fichier.

*Le déplacement de fichiers ne peut actuellement se produire qu'à partir de l'extérieur d'un [navigable](#) , par exemple à partir d'une application de gestionnaire de système de fichiers.*

Si le glissement est initié en dehors de l'application, l'agent utilisateur doit ajouter des éléments à la [liste d'éléments du magasin de données de glissement](#) en fonction des données glissées, en respectant les conventions



de la plate-forme le cas échéant ; cependant, si les conventions de la plate-forme n'utilisent pas [les types MIME](#) pour étiqueter les données glissées, l'agent utilisateur doit faire de son mieux pour mapper les types aux types MIME et, dans tous les cas, toutes les [chaînes de type d'élément de données glissées](#) doivent être [converties en minuscules ASCII](#) .

Les agents utilisateurs peuvent également ajouter un ou plusieurs éléments représentant la sélection ou le(s) élément(s) déplacé(s) sous d'autres formes, par exemple en HTML.

6. Si la [liste des nœuds glissés](#) n'est pas vide, [extrayez les microdonnées de ces nœuds dans un formulaire JSON](#) et ajoutez un élément à la [liste d'éléments du magasin de données glissées](#) , avec ses propriétés définies comme suit :

#### **La chaîne de type d'élément de données glisser**

[application/microdata+json](#)

#### **Le type d'élément de données glisser**

*Texte*

#### **Les données réelles**

La chaîne JSON résultante.

7. Exécutez les sous-étapes suivantes :

1. Soit *urls* une liste vide d' [URL absolues](#) .
2. Pour chaque *nœud* de la [liste des nœuds glissés](#) :

#### **Si le nœud est un [a](#)élément avec un [href](#)attribut**

Ajoutez aux *URL* le résultat de [l'analyse](#) de l'attribut content de l'élément [href](#) par rapport au [nœud document](#) de l'élément .

#### **Si le nœud est un [img](#)élément avec un [src](#) attribut**

Ajoutez aux *URL* le résultat de [l'analyse](#) de l'attribut content de l'élément [src](#) par rapport au [nœud document](#) de l'élément .

3. Si *urls* est toujours vide, alors revenez.
4. Soit *url string* le résultat de la concaténation des chaînes dans *urls* , dans l'ordre où elles ont été ajoutées, séparées par une paire de caractères U+000D CARRIAGE RETURN U+000A LINE FEED (CRLF).
5. Ajoutez un élément à la [liste d'éléments du magasin de données de glissement](#) , avec ses propriétés définies comme suit :

#### **La chaîne de type d'élément de données glisser**

[text/uri-list](#)

#### **Le type d'élément de données glisser**

*Texte*

## Les données réelles

### chaîne d'URL

8. Mettez à jour le [retour par défaut du magasin de données de glissement](#) en fonction de l'agent utilisateur (si l'utilisateur fait glisser la sélection, alors la sélection sera probablement la base de ce retour ; si l'utilisateur fait glisser un élément, alors le rendu de cet élément sera utilisé ; si le glissement a commencé en dehors de l'agent utilisateur, les conventions de la plate-forme pour déterminer le retour de glissement doivent être utilisées).
9. [Déclenchez un événement NPD](#) nommé [dragstart](#) sur le [nœud source](#) .

Si l'événement est annulé, l'opération de glisser-déposer ne doit pas se produire ; retour.

*Étant donné que les événements sans écouteur d'événements enregistrés ne sont, presque par définition, jamais annulés, le glisser-déposer est toujours disponible pour l'utilisateur si l'auteur ne l'empêche pas spécifiquement.*

10. [Déclenchez un événement de pointeur](#) sur le [nœud source](#) nommé [pointercancel](#), et déclenchez tout autre événement de suivi requis par *Pointer Events* . [\[POINTERÉVÉNEMENTS\]](#)
11. [Lancez l'opération de glisser-déposer](#) d'une manière conforme aux conventions de la plate-forme et comme décrit ci-dessous.

Les commentaires par glisser-déposer doivent être générés à partir de la première des sources disponibles suivantes :

1. [Glisser](#) le bitmap du magasin de données , le cas échéant. Dans ce cas, la [coordonnée du point chaud du magasin de données de glissement](#) doit être utilisée comme indice pour savoir où placer le curseur par rapport à l'image résultante. Les valeurs sont exprimées sous forme de distances en [pixels CSS](#) à partir du côté gauche et du côté supérieur de l'image respectivement. [\[CSS\]](#)
2. La [rétroaction par défaut du magasin de données de glissement](#) .

À partir du moment où l'agent utilisateur doit **initier l'opération de glisser-déposer** jusqu'à la fin de l'opération de glisser-déposer, les événements d'entrée de périphérique (par exemple, les événements de souris et de clavier) doivent être supprimés.

Lors de l'opération de glisser, l'élément directement désigné par l'utilisateur comme cible de dépôt est appelé la **sélection immédiate de l'utilisateur** . (Seuls les éléments peuvent être sélectionnés par l'utilisateur ; les autres nœuds ne doivent pas être rendus disponibles en tant que cibles de dépôt.) Cependant, la [sélection immédiate de l'utilisateur](#) n'est pas nécessairement l' **élément cible actuel** , qui est l'élément actuellement sélectionné pour la partie de dépôt du glisser-déposer. et-déposer opération.

La [sélection immédiate de l'utilisateur](#) change lorsque l'utilisateur sélectionne différents éléments (soit en les pointant avec un dispositif de pointage, soit en les sélectionnant d'une autre manière). L' [élément cible actuel](#) change lorsque la [sélection immédiate de l'utilisateur](#) change, en fonction des résultats des écouteurs d'événement dans le document, comme décrit ci-dessous.

L' [élément cible actuel](#) et la [sélection immédiate de l'utilisateur](#) peuvent être nuls, ce qui signifie qu'aucun élément cible n'est sélectionné. Ils peuvent également être tous deux des éléments d'autres documents (basés sur DOM) ou d'autres programmes (non Web). (Par exemple, un utilisateur peut faire glisser du texte vers un traitement de texte.) L' [élément cible courant](#) est initialement nul.

En outre, il existe également une **opération de glissement en cours**, qui peut prendre les valeurs " [none](#)", " [copy](#)", " [link](#)" et " [move](#)". Initialement, il a la valeur " [none](#)". Il est mis à jour par l'agent utilisateur comme décrit dans les étapes ci-dessous.

Les agents utilisateurs doivent, dès que l'opération de glisser est [lancée](#) et toutes les 350 ms ( $\pm$  200 ms) par la suite tant que l'opération de glisser est en cours, [mettre en file d'attente une tâche](#) pour effectuer les étapes suivantes dans l'ordre :

1. Si l'agent utilisateur est toujours en train d'effectuer l'itération précédente de la séquence (le cas échéant) lorsque l'itération suivante arrive à échéance, revenez pour cette itération (en fait, "sauter les images manquées" de l'opération de glisser-déposer).
2. [Déclenchez un événement NPD](#) nommé [drag](#) sur le [nœud source](#). Si cet événement est annulé, l'agent utilisateur doit définir l' [opération de glissement en cours](#) sur " [none](#)" (pas d'opération de glissement).
3. Si l' [drag](#) événement n'a pas été annulé et que l'utilisateur n'a pas terminé l'opération de glisser-déposer, vérifiez l'état de l'opération de glisser-déposer, comme suit :
  1. Si l'utilisateur indique une [sélection utilisateur immédiate](#) différente de celle lors de la dernière itération (ou s'il s'agit de la première itération), et si cette [sélection utilisateur immédiate](#) n'est pas la même que l' [élément cible courant](#), alors mettez à jour l' [élément cible courant](#) comme suit :

**Si la nouvelle [sélection immédiate de l'utilisateur](#) est nulle**

Définissez également l' [élément cible actuel](#) sur null.

**Si la nouvelle [sélection d'utilisateur immédiat](#) se trouve dans un document ou une application non DOM**

Définissez l' [élément cible actuel](#) sur la [sélection immédiate de l'utilisateur](#).

**Sinon**

Lancez un événement NPD nommé dragenter à la sélection immédiate de l'utilisateur .

Si l'événement est annulé, définissez l' élément cible actuel sur la sélection immédiate de l'utilisateur .

Sinon, exécutez l'étape appropriée dans la liste suivante :

Si la sélection immédiate de l'utilisateur est un contrôle de texte (par exemple, textarea ou un input élément dont type l'attribut est à l'état Texte ) ou un hôte d'édition ou un élément modifiable , et que la liste d'éléments du magasin de données de glissement contient un élément avec la chaîne de type d'élément de données de glissement " text/plain " et le texte de type de l'élément de données glisser

Définissez quand même l' élément cible actuel sur la sélection immédiate de l'utilisateur .

Si la sélection immédiate de l'utilisateur est l'élément body

Laissez l' élément cible actuel inchangé.

**Sinon**

Lancez un événement NPD nommé dragenter sur l'élément body , s'il y en a un, ou sur l' Document objet, sinon. Ensuite, définissez l' élément cible actuel sur l'élément body , que cet événement ait été annulé ou non.

2. Si l'étape précédente a entraîné la modification de l' élément cible actuel et si l'élément cible précédent n'était pas nul ou ne faisait pas partie d'un document non DOM, déclenchez un événement NPD nommé dragleave au niveau de l'élément cible précédent, avec le nouvel élément cible actuel comme la *cible connexe* spécifique .
3. Si l' élément cible actuel est un élément DOM, déclenchez un événement NPD nommé dragover à cet élément cible actuel .

Si l' dragover événement n'est pas annulé, exécutez l'étape appropriée dans la liste suivante :

Si l' élément cible actuel est un contrôle de texte (par exemple, textarea ou un input élément dont type l'attribut est à l'état Texte ) ou un hôte d'édition ou un élément modifiable , et que la liste d'éléments du magasin de données de glissement contient un élément avec la chaîne de type d'élément de données de glissement " text/plain " et le texte de type de l'élément de données glisser

Définissez l' opération de glissement actuelle sur " copy " ou " move ", selon les conventions de la plate-forme.

**Sinon**

Réinitialisez l' opération de glissement actuelle sur " none ".

Sinon (si l' dragover événement est annulé), définissez l' opération de glissement actuelle en fonction des valeurs des attributs effectAllowed et de l'objet de l'objet tels qu'ils étaient après la fin de l'envoi de l'événement , conformément au tableau suivant : dropEffectDragEventDataTransfer

<u>effectAllowed</u>	<u>dropEffect</u>	Faites glisser l'opération
" <u>uninitialized</u> ", " <u>copy</u> ", " <u>copyLink</u> ", " <u>copyMove</u> " ou " <u>all</u> "	" <u>copy</u> "	" <u>copy</u> "
" <u>uninitialized</u> ", " <u>link</u> ", " <u>copyLink</u> ", " <u>linkMove</u> " ou " <u>all</u> "	" <u>link</u> "	" <u>link</u> "
" <u>uninitialized</u> ", " <u>move</u> ", " <u>copyMove</u> ", " <u>linkMove</u> " ou " <u>all</u> "	" <u>move</u> "	" <u>move</u> "
Tout autre cas		" <u>none</u> "

4. Sinon, si l' élément cible actuel n'est pas un élément DOM, utilisez des mécanismes spécifiques à la plate-forme pour déterminer quelle opération de glissement est effectuée (aucune, copier, lier ou déplacer) et définissez l' opération de glissement actuelle en conséquence.
5. Mettez à jour le retour de glissement (par exemple, le curseur de la souris) pour qu'il corresponde à l' opération de glissement en cours , comme suit :

Faites glisser l'opération	Retour
" <u>copy</u> "	Les données seront copiées si elles sont déposées ici.
" <u>link</u> "	Les données seront liées si elles sont déposées ici.
" <u>move</u> "	Les données seront déplacées si elles sont déposées ici.
" <u>none</u> "	Aucune opération autorisée, déposer ici annulera l'opération de glisser-déposer.

4. Sinon, si l'utilisateur a terminé l'opération de glisser-déposer (par exemple en relâchant le bouton de la souris dans une interface de glisser-déposer pilotée par la souris), ou si l'événement a été drag annulé, ce sera la dernière itération. Exécutez les étapes suivantes, puis arrêtez l'opération de glisser-déposer :
  1. Si l' opération de glisser en cours est " none" (pas d'opération de glisser), ou si l'utilisateur a terminé l'opération de glisser-déposer en l'annulant (par exemple en appuyant sur la Escape touche ), ou si l' élément cible en cours est nul, alors le glisser l'opération a échoué. Exécutez ces sous-étapes :

1. Soit *dropped* faux.

2. Si l' élément cible actuel est un élément DOM, déclenchez un événement NPDdragleave qui lui est nommé ; sinon, s'il n'est pas nul, utilisez les conventions spécifiques à la plate-forme pour l'annulation du glissement.
3. Définissez l' opération de glissement actuelle sur "none".

Sinon, l'opération glisser pourrait être un succès ; exécutez ces sous-étapes :

4. Let *drop* soit vrai.
5. Si l' élément cible actuel est un élément DOM, déclenchez un événement NPDdrop qui lui est nommé ; sinon, utilisez les conventions spécifiques à la plate-forme pour indiquer une baisse.
6. Si l'événement est annulé, définissez l' opération de glissement en cours sur la valeur de l' dropEffectattribut de l' objet DragEventde l'objet dataTransfer tel qu'il se présentait après la fin de l'envoi de l'événement .

Sinon, l'événement n'est pas annulé ; exécute l'action par défaut de l'événement, qui dépend de la cible exacte comme suit :

**Si l' élément cible actuel est un contrôle de texte (par exemple, textarea ou un inputélément dont typel'attribut est à l' état Texte ) ou un hôte d'édition ou un élément modifiable , et que la liste d'éléments du magasin de données de glissement contient un élément avec la chaîne de type d'élément de données de glissement "text/plain" et le texte de type de l'élément de données glisser**

Insérez les données réelles du premier élément dans la liste des éléments du magasin de données de glissement pour avoir une chaîne de type d'élément de données de glissement de "text/plain" et un type d'élément de données de glissement qui est *du texte* dans le contrôle de texte ou l'hôte d'édition ou l'élément modifiable d'une manière cohérente avec conventions spécifiques à la plate-forme (par exemple, l'insérer à la position actuelle du curseur de la souris ou l'insérer à la fin du champ).

### **Sinon**

Réinitialisez l' opération de glissement actuelle sur "none".

2. Déclenchez un événement NPD nommé dragendsur le nœud source .
3. Exécutez les étapes appropriées de la liste suivante comme action par défaut de l' dragendévénement :

Si *drop* est vrai, l' [élément cible actuel](#) est un *contrôle de texte* (voir ci-dessous), l' [opération de glissement actuelle](#) est "[move](#)", et la source de l'opération de glisser-déposer est une sélection dans le DOM qui est entièrement contenue dans un fichier [d'édition héberger](#)

[Supprimer la sélection](#) .

Si *drop* est vrai, l' [élément cible actuel](#) est un *contrôle de texte* (voir ci-dessous), l' [opération de glissement en cours](#) est "[move](#)", et la source de l'opération de glisser-déposer est une sélection dans un *contrôle de texte*

L'agent utilisateur doit supprimer la sélection glissée du contrôle de texte pertinent.

Si *drop* est faux ou si l' [opération de glissement en cours](#) est "[none](#)"

La drague a été annulée. Si les conventions de la plate-forme exigent que cela soit représenté à l'utilisateur (par exemple en animant la sélection glissée en revenant à la source de l'opération de glisser-déposer), faites-le.

### **Sinon**

L'événement n'a pas d'action par défaut.

Pour les besoins de cette étape, un *contrôle de texte* est un [textarea](#)élément ou un [input](#)élément dont [type](#)l'attribut est dans l'un des états [Text](#) , [Search](#) , [Tel](#) , [URL](#) , [Email](#) , [Password](#) ou [Number](#) .

*Les agents utilisateurs sont encouragés à réfléchir à la façon de réagir aux glissements près du bord des régions défilantes. Par exemple, si un utilisateur fait glisser un lien vers le bas de la [fenêtre d'affichage](#) sur une longue page, il peut être judicieux de faire défiler la page afin que l'utilisateur puisse déposer le lien plus bas sur la page.*

*Ce modèle est indépendant de [Document](#)l'objet d'où proviennent les nœuds impliqués ; les événements sont déclenchés comme décrit ci-dessus et le reste du modèle de traitement s'exécute comme décrit ci-dessus, quel que soit le nombre de documents impliqués dans l'opération.*

## **6.10.6 Résumé des événements**

*Cette section est non normative.*

Les événements suivants sont impliqués dans le modèle glisser-déposer.



Nom de l'événement	Cible	Annulable ?	Faites glisser le mode magasin de données	<u>dropEffect</u>	Action par défaut
<u>dragstart</u> ✓ MDN	<u>Nœud source</u>	✓ Annulable	<u>Mode lecture/écriture</u>	" <u>none</u> "	Lancer l'opération de glisser-déposer
<u>drag</u> ✓ MDN	<u>Nœud source</u>	✓ Annulable	<u>Mode protégé</u>	" <u>none</u> "	Continuez l'opération de glisser-déposer
<u>dragenter</u> ✓ MDN	<u>Sélection immédiate de l'utilisateur ou de l'élément body</u>	✓ Annulable	<u>Mode protégé</u>	<u>Basé sur effectAllowed la valeur</u>	Rejeter la sélection immédiate de l'utilisateur en tant qu'élément cible potentiel
<u>dragleave</u> ✓ MDN	<u>Élément cible précédent</u>	—	<u>Mode protégé</u>	" <u>none</u> "	Aucun
<u>dragover</u> ✓ MDN	<u>Élément cible actuel</u>	✓ Annulable	<u>Mode protégé</u>	<u>Basé sur effectAllowed la valeur</u>	Réinitialiser l'opération de glissement actuelle sur "aucune"
<u>drop</u> ✓ MDN	<u>Élément cible actuel</u>	✓ Annulable	<u>Mode lecture seule</u>	<u>Opération de glissement en cours</u>	Varie
<u>dragend</u> ✓ MDN	<u>Nœud source</u>	—	<u>Mode protégé</u>	<u>Opération de glissement en cours</u>	Varie

Tous ces événements bouillonnent, sont composés et l' effectAllowed attribut a toujours la valeur qu'il avait après l' dragstart événement, par défaut "uninitialized" dans l' dragstart événement.

### 6.10.7 L' draggable attribut



Tous [les éléments HTML](#) peuvent avoir l' **draggable**attribut content défini. L' **draggable**attribut est un [attribut énuméré](#) . Il a trois états. Le premier état est *vrai* et il a le mot-clé `true`. Le deuxième état est *faux* et contient le mot-clé `false`. Le troisième état est *auto* ; il n'a pas de mots clés mais c'est la [valeur manquante default](#) et la [valeur invalide default](#) .

L' état *vrai* signifie que l'élément est déplaçable ; l' état *faux* signifie qu'il ne l'est pas. L' état *automatique* utilise le comportement par défaut de l'agent utilisateur.

Un élément avec un **draggable**attribut doit également avoir un **title**attribut qui nomme l'élément dans le but d'interactions non visuelles.

**element.draggable [ = value ]**

Renvoie `true` si l'élément est déplaçable ; sinon, renvoie `faux`.

Peut être défini pour remplacer la valeur par défaut et définir l' **draggable**attribut de contenu.

L' **draggable**attribut IDL, dont la valeur dépend de l'attribut de contenu de la manière décrite ci-dessous, contrôle si l'élément est déplaçable ou non. Généralement, seules les sélections de texte sont déplaçables, mais les éléments dont **draggable**l'attribut IDL est vrai deviennent également déplaçables.

Si l'attribut content d'un élément **draggable**a l'état *true* , l' **draggable**attribut IDL doit renvoyer `true`.

Sinon, si l' **draggable**attribut content de l'élément a l'état *false* , l' **draggable**attribut IDL doit renvoyer `false`.

Sinon, l' **draggable**attribut content de l'élément a l'état *auto* . Si l'élément est un **img**élément, un **object** élément qui [représente](#) une image ou un **a**élément avec un **href**attribut de contenu, l' **draggable**attribut IDL doit renvoyer `true` ; sinon, l' **draggable**attribut IDL doit retourner `false`.

Si l' **draggable**attribut IDL est défini sur la valeur `false`, l' **draggable**attribut content doit être défini sur la valeur littérale " `false`". Si l' **draggable**attribut IDL est défini sur la valeur `true`, l' **draggable**attribut content doit être défini sur la valeur littérale " `true`".

### 6.10.8 Risques de sécurité dans le modèle glisser-déposer

Les agents utilisateurs ne doivent pas mettre les données ajoutées à l' **DataTransfer**objet pendant l' **dragstart**événement à la disposition des scripts jusqu'à l' **drop**événement, car sinon, si un utilisateur faisait glisser des informations sensibles d'un document vers un deuxième document, traversant un troisième

document hostile dans le processus, le document hostile pourrait intercepter les données.

Pour la même raison, les agents utilisateurs doivent considérer qu'un dépôt est réussi uniquement si l'utilisateur a spécifiquement mis fin à l'opération de glisser - si des scripts mettent fin à l'opération de glisser, il doit être considéré comme un échec (annulé) et l'événement ne doit pas être déclenché [drop](#).

Les agents utilisateurs doivent veiller à ne pas lancer d'opérations de glisser-déposer en réponse à des actions de script. Par exemple, dans un environnement de souris et de fenêtre, si un script déplace une fenêtre alors que l'utilisateur a le bouton de la souris enfoncé, l'UA ne considérera pas cela comme le démarrage d'un glissement. Ceci est important car sinon, les UA pourraient faire en sorte que des données soient extraites de sources sensibles et déposées dans des documents hostiles sans le consentement de l'utilisateur.

Les agents utilisateurs doivent filtrer le contenu potentiellement actif (scripté) (par exemple, HTML) lorsqu'il est glissé et lorsqu'il est déposé, en utilisant une liste sécurisée de fonctionnalités sûres connues. De même, [les URL relatives](#) doivent être transformées en URL absolues pour éviter que les références ne changent de manière inattendue. Cette spécification ne précise pas comment cela est effectué.

Considérez une page hostile fournissant du contenu et obligeant l'utilisateur à sélectionner et glisser-déposer (ou même copier-coller) ce contenu dans la [contenteditable](#) région d'une page victime. Si le navigateur ne garantit pas que seul le contenu sûr est déplacé, le contenu potentiellement dangereux tel que les scripts et les gestionnaires d'événements dans la sélection, une fois déposé (ou collé) dans le site victime, obtient les privilèges du site victime. Cela permettrait ainsi une attaque par cross-site scripting.

## 6.11 L' [popover](#)attribut

Tous [les éléments HTML](#) peuvent avoir l' [popover](#)attribut content défini. Lorsqu'il est spécifié, l'élément ne sera pas rendu tant qu'il ne sera pas affiché, auquel cas il sera rendu au-dessus du contenu d'une autre page.

L' [popover](#)attribut est un [attribut énuméré](#) . Le tableau suivant répertorie les états de cet attribut :

État	Mots clés	Description
État automatique	<a href="#">auto</a>	Ferme les autres popovers lorsqu'ils sont ouverts ; a <a href="#">rejeter la lumière</a> .
	La chaîne vide	
État manuel	<a href="#">manual</a>	Ne ferme pas les autres popovers ; ne <a href="#">s'allume pas</a> .

L'attribut peut être omis. La [valeur par défaut invalide](#) est l' [état manuel](#) . La [valeur manquante par défaut](#) est l' **état sans popover** .

L' **popover** attribut IDL doit [réfléter](#) l' attribut [popover](#) , [limité aux seules valeurs connues](#) .

Chaque [élément HTML](#) a un **état de visibilité popover** , initialement [hidden](#) , avec ces valeurs potentielles :

- **caché**
- **montrant**

Le [Document](#) a une **liste popover automatique** , qui est une [liste](#) , initialement vide.

Le [Document](#) a un **popover pointerdown target** , qui est un [élément HTML](#) ou null, initialement null.

Chaque [élément HTML](#) a un **invocateur de popover** , qui est un [élément HTML](#) ou null, initialement défini sur null.

Chaque [élément HTML](#) a une **tâche de basculement popover** , initialement null, qui est soit null, soit une [structure](#) qui a :

### **Tâche**

Une tâche qui déclenche un [ToggleEvent](#).

### **Ancien état**

Une chaîne qui représente la valeur de l'événement de la [tâcheoldState](#) pour l' attribut.

[Les étapes de changement d'attribut](#) suivantes sont utilisées pour tous [les éléments HTML](#) :

1. Si *l'espace de noms* n'est pas nul, alors retournez.
2. Si *localName* n'est pas [popover](#), alors retournez.
3. Si *oldValue* et *value* sont dans [des états](#) différents , exécutez l' [algorithme de masquage du popover](#) avec *element* , true, true et false.

### **`element.showPopover()`**

Affiche l' *élément* popover en l'ajoutant au calque supérieur. Si l'attribut de l' *élément* [popover](#) est dans l' [état automatique](#) , cela fermera également tous les autres popovers [automatiques à moins qu'ils ne soient un ancêtre de l'élément](#) selon l' algorithme [d'ancêtre du popover le plus élevé](#) .

### **`element.hidePopover()`**

Masque l' *élément* popover en le supprimant du calque supérieur et `display:none` en l'appliquant.

**`element.togglePopover()`**

Si l' *élément* popover ne s'affiche pas, cette méthode l'affiche. Sinon, cette méthode le masque.

Les **`showPopover()`** étapes de la méthode sont :

1. Exécutez [show popover](#) étant donné [ceci](#) et vrai.

Pour **afficher popover** , étant donné un *élément* [d'élément HTML](#) et un booléen *throwExceptions* :

1. Si le résultat de l'exécution [de la vérification de la validité du popover de l'élément](#) donné et de *false* est faux, alors :
  1. [Assert](#) : *throwExceptions* est vrai.
  2. Lancez un ["InvalidStateError" DOMException](#) .
2. [Assert](#) : l'*élément* n'est pas dans [la couche supérieure](#) du [document](#) du nœud de l' *élément* .
3. Si le résultat du [déclenchement d'un événement](#) nommé [beforetoggle](#), à l'aide de [ToggleEvent](#), avec l' [cancelable](#)attribut initialisé à vrai, l' [oldState](#)attribut initialisé à `"closed"` et l' [newState](#) attribut initialisé à `"open"` à l'*élément* est faux, alors retournez.
4. Si le résultat de l'exécution [de la vérification de la validité du popover de l'élément](#) donné et faux est faux :
  1. Si *throwExceptions* est vrai, lancez un ["InvalidStateError" DOMException](#) .
  2. Sinon, reviens.

[La vérification de la validité du popover](#) est appelée à nouveau car le [déclenchement de l' beforetoggle](#)événement peut avoir déconnecté cet élément ou modifié son [popover](#)attribut.

5. Soit *document* le [nœud document](#) de l' *élément* .
6. Soit *shouldRestoreFocus* être faux.
7. Si l'attribut de l' *élément* [popover](#) est dans l' [état auto](#) , alors :
  1. Soit *originalType* la valeur de l'attribut de l' *élément* [popover](#) .
  2. Soit *ancêtre* le résultat de l'exécution de l'algorithme [d'ancêtre popover le plus élevé](#) donné *element* .
  3. Exécutez [masquer tous les popovers jusqu'à l'ancêtre](#) donné , faux et vrai.

4. Si *originalType* n'est pas égal à la valeur de l'attribut de l'élément `popover` , ou si le résultat de l'exécution de la vérification de la validité du popover de l'élément donné et `false` est faux :

1. Si *throwExceptions* est vrai, lancez un `"InvalidStateError" DOMException` .
2. Sinon, reviens.

*Vérifier la validité du popover est appelé à nouveau car l'exécution de masquer tous les popovers jusqu'à ce que ci-dessus aurait pu déclencher l' `beforetoggle` événement, et un gestionnaire d'événement aurait pu déconnecter cet élément ou appeler `showPopover()` cet élément.*

8. Si le résultat de l'exécution du popover automatique le plus élevé sur le *document* est nul, définissez *shouldRestoreFocus* sur `true`.

*Cela garantit que le focus est renvoyé à l'élément précédemment focalisé uniquement pour le premier popover d'une pile.*

9. Ajouter un élément à la liste contextuelle automatique du *document* .
10. Définissez l'élément précédemment focalisé de l'élément sur `null`.
11. Soit *originalFocusedElement* la zone focalisée du *document* de l'ancre DOM du *document* .
12. Ajouter un élément au calque supérieur du *document* .
13. Définissez l'état de visibilité du popover de l'élément sur Affichage .
14. Exécutez les étapes de mise au point popover élément donné .
15. Si *shouldRestoreFocus* est vrai et que l'attribut de l'élément `popover` n'est pas dans l' état sans popover , alors définissez l'élément précédemment focalisé de l'élément sur *originFocusedElement* .
16. Mettre en file d'attente une tâche d'événement de basculement de popover en fonction de l'élément , `"closed"` et `"open"`.

Pour **mettre en file d'attente une tâche d'événement de basculement popover** en fonction d'un élément *element* , d'une chaîne *oldState* et d'une chaîne *newState* :

1. Si la tâche de basculement popover de l'élément n'est pas nulle, alors :
  1. Définissez *oldState* sur l'ancien état de la tâche de basculement popover de l'élément .
  2. Supprimez la tâche de basculement de la fenêtre contextuelle de l'élément de sa file d'attente de tâches .

3. Définissez [la tâche de basculement popover](#) de l' *élément* sur null.
2. [Mettez en file d'attente une tâche d'élément](#) en fonction de [la source de la tâche d'interaction utilisateur](#) et de l'*élément* pour exécuter les étapes suivantes :
  1. [déclencher un événement](#) nommé `toggle`, using `ToggleEvent`, avec l'`oldState`attribut initialisé à `oldState` et l' `newState`attribut initialisé à `newState` à *element* .
  2. Définissez [la tâche de basculement popover](#) de l' *élément* sur null.
  3. Définissez [la tâche de basculement contextuelle](#) de l'*élément* sur une structure avec [la tâche](#) définie sur la [tâche](#) qui vient d'être mise en file d'attente et l'[ancien état](#) défini sur `oldState` .

Les `hidePopover()` étapes de la méthode sont :

1. Exécutez l' [algorithme de masquage du popover](#) étant donné `this` , `true`, `true` et `true`.

Pour **masquer un popover** étant donné un *élément* [d'élément HTML](#) , un booléen `focusPreviousElement` , un booléen `fireEvents` et un booléen `throwExceptions` :

1. Si le résultat de l'exécution [de la vérification de la validité du popover de l'élément](#) donné et vrai est faux :
  1. Si `throwExceptions` est vrai, lancez un `"InvalidStateError" DOMException` .
  2. Sinon, reviens.
2. Soit *document* le [nœud document](#) de l' *élément* .
3. Si l'attribut de l' *élément* `popover` est dans l' [état auto](#) , alors :
  1. Exécutez [masquer tous les popovers jusqu'à l'élément](#) donné , `focusPreviousElement` et `fireEvents` .
  2. Si l'*élément* n'est pas dans [la liste déroulante automatique](#) du *document* :
    1. Si `throwExceptions` est vrai, lancez un `"InvalidStateError" DOMException` .
    2. Sinon, reviens.
  3. [Assert](#) : Le dernier élément de [la liste déroulante automatique](#) du *document* est *element* .
  4. [Supprimer](#) l'*élément* de [la liste déroulante automatique](#) du *document* .

4. Définissez l'invocateur de popover de l'élément sur null.
5. Si *fireEvents* est vrai :
  1. Lancez un événement nommé beforetoggle, en utilisant ToggleEvent, avec l' oldState attribut initialisé à " open" et l' newState attribut initialisé à " closed" à l'élément .
  2. Si le résultat de l'exécution de la vérification de la validité du popover de l'élément donné et vrai est faux :
    1. Si *throwExceptions* est vrai, lancez un " InvalidStateError" DOMException .
    2. Sinon, reviens.

La vérification de la validité du popover est appelée à nouveau car le déclenchement de l' beforetoggle événement peut avoir déconnecté cet élément ou modifié son popover attribut.

6. Supprimer l'élément de la couche supérieure .
7. Définissez l'état de visibilité du popover de l' élément sur masqué .
8. Si *fireEvents* est vrai, alors mettez en file d'attente une tâche d'événement de basculement popover donnée *element* , " open" et " closed".
9. Soit *previousFocusedElement* l' élément précédemment focalisé de l'élément .
10. Si *previousFocusedElement* n'est pas nul, alors :
  1. Définissez l'élément précédemment focalisé de l'élément sur null.
  2. Si *focusPreviousElement* est vrai, alors exécutez les étapes de mise au point pour *previousFocusedElement* ; la fenêtre ne doit pas défiler en effectuant cette étape.

Les étapes de la méthode sont : togglePopover (force)

1. Si l'état de visibilité de ce popover s'affiche et que la *force* n'est pas présente ou fausse, exécutez l' algorithme de masquage du popover en fonction de this , true, true et true.
2. Sinon, si *la force* n'est pas présente ou vraie, alors exécutez show popover étant donné ceci et vrai.

Pour **masquer tous les popovers jusqu'à** , étant donné un *point de terminaison* d'élément HTML , un booléen *focusPreviousElement* et un booléen *fireEvents* :

1. Soit *document* le nœud document du *point de terminaison* .



2. Soit *closeAllOpenPopovers* un algorithme qui effectue les étapes suivantes :
  1. Laissez *popover* être [le popover automatique le plus haut](#) du document .
  2. Tant que *popover* n'est pas nul :
    1. Exécutez l' [algorithme de masquage du popover](#) avec *popover* , *focusPreviousElement* , *fireEvents* et *false*.
    2. Définissez *le popover* sur [le popover automatique le plus élevé](#) du document .
3. Si le point de terminaison est nul, exécutez *closeAllOpenPopovers* et revenez.
4. Soit *lastToHide* égal à null.
5. Laissez *foundEndpoint* être faux.
6. Pour chaque *popover* dans [la liste des popovers automatiques](#) du document :
  1. Si *popover* est *endpoint* , définissez *foundEndpoint* sur *true*.
  2. Sinon, si *foundEndpoint* est *true*, définissez *lastToHide* sur *popover* et [break](#) .
7. Si *foundEndpoint* est faux, exécutez *closeAllOpenPopovers* et revenez.
8. Alors que *lastToHide* n'est pas null et que [l'état de visibilité du popover](#) de *lastToHide* est [affiché](#) et que [la liste de popover automatique](#) du document n'est pas vide :
  1. Exécutez l' [algorithme de masquage de popover](#) en fonction du dernier élément de [la liste de popover automatique](#) du document , *focusPreviousElement* , *fireEvents* et *false*.

*L' [algorithme masquer tous les popovers jusqu'à](#) est utilisé dans plusieurs cas pour masquer tous les popovers qui ne restent pas ouverts lorsque quelque chose se produit. Par exemple, lors du *light-dismiss* d'un popover, cet algorithme garantit que nous ne fermons que les popovers qui ne sont pas liés au nœud cliqué par l'utilisateur.*

Pour trouver l' **ancêtre du popover le plus élevé** , étant donné un [Node](#) *newPopover* , procédez comme suit. Ils renvoient un [élément HTML](#) ou null.

*L' [algorithme d'ancêtre du popover le plus haut](#) renverra le popover ancêtre le plus haut (le plus haut dans la [liste des popovers automatiques](#) ) pour le popover fourni. Les popovers peuvent être liés les uns aux autres de plusieurs manières, créant ainsi une arborescence de popovers. Il existe deux chemins par lesquels un popover (appelez-le le popover "enfant") peut avoir un popover ancêtre (appelez-le le popover "parent") :*



1. Les popovers sont imbriqués les uns dans les autres dans l'arborescence DOM. Dans ce cas, le popover descendant est "l'enfant" et son popover ancêtre est le "parent".
2. Un élément invoquant (par exemple a [button](#)) a l'un des attributs invoquants (par exemple [popovertoggletarget](#)) pointant vers un popover. Dans ce cas, le popover est "l'enfant" et le popover contenu dans le DOM de l'élément appelant est le "parent". L'invocateur doit être dans un popover et référencer un popover ouvert.

Dans chacune des relations formées ci-dessus, le popover parent doit être strictement inférieur dans la [liste des popovers automatiques](#) au popover enfant, sinon il ne forme pas une relation ancestrale valide. Cela élimine les popovers non affichés et les auto-pointeurs (par exemple, un popover avec un attribut d'ancrage qui pointe vers le même popover), et cela permet la construction d'un arbre bien formé à partir du graphe (éventuellement cyclique) des connexions. Par exemple, si deux popovers ont des ancres pointant l'une vers l'autre, la seule relation valide est que la première à s'ouvrir est le "parent" et la seconde est l'"enfant". Seuls les popovers [automatiques](#) sont pris en compte.

1. Soit `popoverPositions` une [carte](#) vide .
2. Soit `indice` 0.
3. Soit `document` le [nœud document](#) de `newPopover` .
4. Pour chaque `popover` dans [la liste des popovers automatiques](#) du `document` :
  1. [Définissez](#) `popoverPositions [ popover ]` sur `index` .
  2. Incrémenter `l'indice` de 1.
5. [Définissez](#) `popoverPositions [ newPopover ]` sur `index` .
6. Incrémenter `l'indice` de 1.
7. Laissez `topmostPopoverAncestor` être nul.
8. Soit `checkAncestor` un algorithme qui effectue les étapes suivantes étant donné le `candidat` :
  1. Si le `candidat` est nul, alors retournez.
  2. Soit `candidateAncestor` le résultat de l'exécution [du popover ouvert inclusif le plus proche](#) `candidat` donné .
  3. Si `candidateAncestor` est null, alors retournez.
  4. Soit `candidatePosition` être `popoverPositions [ candidateAncestor ]`.

5. Si *topmostPopoverAncestor* est null  
ou *popoverPositions* [ *topmostPopoverAncestor* ] est inférieur  
à *candidatePosition* , alors  
définissez *topmostPopoverAncestor* sur *candidateAncestor* .
9. Exécutez *checkAncestor* en fonction du résultat de l'exécution [du popover ouvert inclusif le plus proche](#) en fonction du nœud parent  
de *newPopover* dans l' [arborescence plate](#) .
10. Pour chaque *invocateur* dans [les invocateurs popover](#) du *document* :
  1. Si [l'élément cible du popover](#) de *l'invocateur* est *newPopover* , alors  
exécutez *checkAncestor* en fonction de *l'invocateur* .
11. renvoie *topmostPopoverAncestor* .

Pour trouver le **popover ouvert inclusif le plus proche** étant donné un [Node](#) *nœud* ,  
procédez comme suit. Ils renvoient un [élément HTML](#) ou null.

1. Soit *currentNode* être *node* .
2. Tant que *currentNode* n'est pas null :
  1. Si l'attribut de *currentNode* [popover](#) est dans l' [état automatique](#) et  
que [l'état de visibilité](#) du popover de *currentNode* est [affiché](#) , alors  
retourne *currentNode* .
  2. Définissez *currentNode* sur le parent de *currentNode* dans  
l' [arborescence plate](#) .
3. Renvoie nul.

Pour **trouver la fenêtre contextuelle automatique la plus élevée** d'un [Document](#) *document* , procédez comme suit. Ils renvoient un [élément HTML](#) ou null.

1. Si [la liste contextuelle automatique](#) du *document* n'est pas vide, renvoie le  
dernier élément de [la liste contextuelle automatique](#) du *document* .
2. Renvoie nul.

Pour effectuer les **étapes de focus popover** pour un *sujet* [d'élément HTML](#) :

1. Soit *control* le [délégué de focus](#) du *sujet* donné " *other* " et vrai.
2. Si *le contrôle* est nul, alors retournez.
3. Exécutez les [étapes de mise au](#) point contrôlées .

4. Soit *topDocument* le [document actif du contexte](#) de navigation de [niveau supérieur](#) du document du [nœud](#) du *contrôle* .
5. Si l'[origine](#) du [document du nœud](#) du *contrôle* n'est pas la [même](#) que l'[origine](#) de *topDocument* , alors retournez.
6. [Vide les candidats autofocus](#) de *topDocument* .
7. Définissez [l'indicateur de traitement de mise au point automatique](#) de *topDocument* sur true.

Pour **vérifier la validité du popover** pour un élément [d'élément HTML](#) avec un booléen *expectToBeShowing* , procédez comme suit. Ils renvoient un booléen.

1. Si l'attribut de l' élément [popover](#) est dans l' [état sans popover](#) , alors renvoie false.
2. Si *element* n'est pas [connecté](#) , alors retourne false.
3. Si *expectToBeShowing* est true et que [l'état de visibilité du popover](#) de l'élément ne [s'affiche](#) pas , alors renvoyez false.
4. Si *expectToBeShowing* est false et que [l'état de visibilité du popover](#) de l'élément n'est pas [hidden](#) , alors renvoyez false.
5. Si *element* est un [dialog](#)élément et a l' [open](#)attribut, alors retourne false.
6. Si [l'indicateur plein écran](#) de l'élément est défini, renvoie false.
7. Renvoie vrai.

### 6.11.1 Les *attributs de la cible popover*

[Les boutons](#) peuvent avoir les attributs de contenu suivants, connus sous le nom d' [attributs de cible de popover](#) :

- [popovertoggletarget](#)
- [popoverhidetarget](#)
- [popovershowtarget](#)

Les [attributs de cible de popover](#) permettent à certains types de boutons d'afficher et de masquer l'élément avec l' [popover](#)attribut. Si un attribut cible de popover est spécifié, la valeur de cet attribut doit être l'ID de l'élément avec l' [popover](#)attribut.

Ce qui suit montre comment [popovershowtarget](#)peut être utilisé pour ouvrir un popover :

```

<div popover=auto id="foo">
  This is a popover!
</div>

<button popovershowtarget="foo">
  Show a popover
</button>

```

Ce qui suit montre comment [popovertoggletarget](#) ouvrir et fermer un popover manuel, qui ne peut pas être fermé avec un léger rejet :

```

<div popover=manual id="foo">
  This is a popover!
</div>

<button popovertoggletarget="foo">
  Show or hide a popover
</button>

```

Interface DOM :

```
interface mixin PopoverTargetElement {
```

```
  [CEReactions] attribute Element? popoverToggleTargetElement;
```

```
  [CEReactions] attribute Element? popoverHideTargetElement;
```

```
  [CEReactions] attribute Element? popoverShowTargetElement;
```

```
};
```

L' **popoverToggleTargetElement** attribut IDL doit [réfléter](#) l' [popovertoggletarget](#) attribut.

L' **popoverHideTargetElement** attribut IDL doit [réfléter](#) l' [popoverhidetarget](#) attribut.

L' **popoverShowTargetElement** attribut IDL doit [réfléter](#) l' [popovershowtarget](#) attribut.

Pour exécuter le **comportement d'activation de l'attribut popover target** étant donné un [Node](#) *nœud* :

1. Soit *popover* l' [élément cible popover](#) du *nœud* .
2. Si *popover* est nul, alors retournez.
3. Si le *nœud* n'a pas l' [popovertoggletarget](#) attribut, alors :

1. Si *le nœud* a un `popovershowtarget`attribut et que l'état de visibilité du *popover* est affiché , alors retournez.
2. Si *le nœud* a un `popoverhidetarget`attribut et que l'état de visibilité du *popover* est masqué , alors retournez.
4. Si l'état de visibilité du *popover* de *popover* s'affiche et que le résultat de l'exécution de la vérification de la validité du popover étant donné que *popover* et true est vrai, exécutez l' algorithme de masquage du popover avec *popover* , true, true et false.
5. Sinon, si l'état de visibilité du *popover* de *popover* est masqué et que le résultat de l'exécution de la vérification de la validité du popover donné *popover* et false est vrai :
  1. Définissez l'invocateur de popover de *popover* sur *node* .
  2. Exécutez show popover avec *popover* et false.

Pour obtenir l' **élément cible du popover** en fonction d'un `Node` *nœud* , procédez comme suit. Ils renvoient un élément HTML ou null.

1. Si *node* n'est pas un bouton , alors retournez null.
2. Si *node* est disabled , alors retournez null.
3. Si *le nœud* a un propriétaire de formulaire et que *le nœud* est un bouton d'envoi , alors renvoyez null.
4. Soit *idref* nul.
5. Si *le nœud* a un `popovertoggletarget`attribut, alors définissez *idref* sur la valeur de l'attribut du *nœud*`popovertoggletarget` .
6. Sinon, si *le nœud* a un `popovershowtarget`attribut, alors définissez *idref* sur la valeur de l'attribut du *nœud*`popovershowtarget` .
7. Sinon, si *le nœud* a un `popoverhidetarget`attribut, alors définissez *idref* sur la valeur de l'attribut du *nœud*`popoverhidetarget` .
8. Si *idref* est null, alors retourne null.
9. Soit *popoverElement* le premier élément dans l'arborescence , parmi les descendants de la racine du *nœud* , dont l'ID est *idref* ; sinon, s'il n'y a pas un tel élément , null.
10. Si *popoverElement* est null, alors retournez null.
11. Si l'attribut de *popoverElement*`popover` est dans l' état no popover , alors renvoyez null.
12. Renvoie *popoverElement* .

Pour obtenir les **invocateurs de popover** donnés à un [Document](#) *document* :

1. Renvoie un *document* [HTMLCollection](#) ancré à dont le filtre correspond à des éléments qui sont [des boutons](#) .

### 6.11.2 Ignorer la lumière Popover

*"Rejeter léger" signifie qu'appuyer sur la `Esc` touche ou cliquer en dehors d'un popover dont `popover` l'attribut est à l'état [automatique](#) fermera le popover.*

**Annulation des popovers** : lorsqu'un [Document](#) popover [automatique le plus élevé](#) s'affiche, les agents utilisateurs peuvent fournir une interface utilisateur qui, lors de l'activation, [met en file d'attente une tâche d'élément](#) sur la [source de la tâche d'interaction de l'utilisateur](#) étant donné le [popover automatique le plus élevé](#) pour exécuter l' [algorithme de masquage du popover](#) étant donné le [popover automatique le plus élevé](#) , true , vrai et faux.

Pour **allumer et ignorer les popovers ouverts** , étant donné un [Event](#) *événement* :

1. [Assert](#) : l'attribut de l' événement [isTrusted](#) est vrai.
2. Soit *cible* la [cible](#) de l'événement .
3. Soit *topmostPopover* le résultat de l'exécution [du popover automatique le plus haut](#) donné *target* .
4. Si *topmostPopover* est null, alors retournez.
5. Soit *document* le [nœud document](#) de la *cible* .
6. Si l'événement est un [PointerEvent](#) et l' événement *type* est [pointerdown](#), alors :
  1. Définissez [la cible du pointeur vers le bas du popover](#) du *document* sur le résultat de l'exécution de la *cible* [popover la plus cliquée](#) et de `".inclusive"`
7. Si l'événement est un [PointerEvent](#) et l' événement *type* est [pointerup](#), alors :
  1. Soit *ancêtre* le résultat de l'exécution [du popover le plus cliqué en fonction de](#) la *cible* donnée .
  2. Laissez *sameTarget* être vrai si *ancestor* est [la cible popover pointerdown](#) du *document* .
  3. Définissez [la cible du pointeur vers le bas du popover](#) du *document* sur null.

4. Si *sameTarget* est true, exécutez [hide all popovers jusqu'à l'ancêtre donné](#) , false et true.

**Les popovers ouverts de rejet léger seront appelés par la [spécification Pointer Events](#) lorsque l'utilisateur clique ou touche n'importe où sur la page.**

Pour trouver le **popover cliqué le plus haut** , étant donné un [Node](#) *nœud* :

1. Soit *clickedPopover* le résultat de l'exécution du *nœud* [popover ouvert inclus le plus proche](#) donné .
2. Soit *invokerPopover* le résultat de l'exécution [du popover cible inclusif le plus proche pour](#) le *nœud* donné par l'invocateur .
3. Soit *getStackPosition* un algorithme qui effectue les étapes suivantes étant donné un *popover* d' [élément HTML](#) :
  1. Soit *popoverList* la [liste popover automatique](#) du [document du](#) *nœud popover* .
  2. Si *popover* est dans *popoverList* , alors retournez l'index de *popover* dans *popoverList* + 1.
  3. Renvoie 0.
4. Si le résultat de l'exécution de *getStackPosition* étant donné *clickedPopover* est supérieur au résultat de l'exécution de *getStackPosition* étant donné *invocatorPopover* , alors retourne *clickedPopover* .
5. Renvoie *invocatorPopover* .

Pour trouver le **popover cible inclusif le plus proche pour l'invocateur** étant donné un [Node](#) *nœud* :

1. Soit *currentNode* être *node* .
2. Tant que *currentNode* n'est pas null :
  1. Soit *targetPopover* l' [élément cible](#) du popover de *currentNode* .
  2. Si *targetPopover* n'est pas null et que l'attribut de *targetPopover* [popover](#) est dans l' [état auto](#) et que [l'état de visibilité](#) du popover de *targetPopover* est [affiché](#) , alors retourne *targetPopover* .
  3. Définissez *currentNode* sur l'ancêtre de *currentNode* dans l' [arborescence plate](#) .

### 6.11.3 L' **ToggleEvent** interface

```
[Exposed=Window]

interface ToggleEvent : Event {

    constructor(DOMString type, optional ToggleEventInit
eventInitDict = {});

    readonly attribute DOMString oldState;

    readonly attribute DOMString newState;

};

dictionary ToggleEventInit : EventInit {

    DOMString oldState = "";

    DOMString newState = "";

};
```

#### **event.oldState**

Réglez sur "closed" lors de la transition de fermé à ouvert, ou sur "open" lors de la transition d'ouvert à fermé.

#### **event.newState**

Réglez sur "open" lors de la transition de fermé à ouvert, ou sur "closed" lors de la transition d'ouvert à fermé.

L' **oldState** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il est initialisé à "open" si l'élément avec l' [état de visibilité popover](#) de l'attribut est [affiché](#) ; sinon ".closed"

L' **newState** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il est initialisé à "closed" si l'élément avec l' [état de visibilité popover](#) de l'attribut est [affiché](#) ; sinon ".open"



## 7 Chargement des pages Web

Cette section décrit les fonctionnalités qui s'appliquent le plus directement aux navigateurs Web. Cela dit, sauf indication contraire, les exigences définies dans cette section *s'appliquent* à tous les agents utilisateurs, qu'ils soient ou non des navigateurs Web.

### 7.1 Notions complémentaires

#### 7.1.1 Origines

Les origines sont la devise fondamentale du modèle de sécurité du Web. Deux acteurs de la plateforme web qui partagent une origine sont supposés se faire confiance et avoir la même autorité. Des acteurs d'origines différentes sont considérés comme potentiellement hostiles les uns aux autres et sont isolés les uns des autres à des degrés divers.

Par exemple, si le site Web d'Example Bank, hébergé sur `bank.example.com`, tente d'examiner le DOM du site Web d'Example Charity, hébergé sur `charity.example.org`, un `"SecurityError" DOMException` sera généré.

---

Une **origine** est l'une des suivantes :

##### Une *origine opaque*

Une valeur interne, sans sérialisation à partir de laquelle elle peut être recrée (elle est sérialisée en tant que `"null"` par [sérialisation d'un origin](#)), pour laquelle la seule opération significative teste l'égalité.

##### Une *origine tuple*

Un [tuple](#) est composé de :

- Un **schéma** (une [chaîne ASCII](#)).
- Un **hôte** (un [hôte](#)).
- Un **port** (null ou un entier non signé 16 bits).
- Un **domaine** (null ou un [domaine](#)). Nul sauf indication contraire.

*Les origines peuvent être partagées, par exemple, entre plusieurs [Document](#) objets. De plus, les origines sont généralement immuables. Seul le [domaine](#) d'une [origine de tuple](#) peut être modifié, et uniquement via l'[document.domain](#) API.*

Le **domaine effectif** d'une *origine* [origine](#) se calcule comme suit :

1. Si l'*origine* est une [origine opaque](#) , alors renvoie null.
2. Si [le domaine](#) de l' *origine* n'est pas nul, alors retournez [le domaine](#) de l' *origine* .
3. Renvoie [l'hôte](#) d' *origine* .

La **sérialisation d'une origine** est la chaîne obtenue en appliquant l'algorithme suivant à l' *origine* [origin](#) donnée :

1. Si l'*origine* est une [origine opaque](#) , alors retourne " `null`".
2. Sinon, laissez *result* être [le schéma](#) d' *origine* .
3. Ajouter " `://`" au *résultat* .
4. Ajouter [l'hôte](#) d' *origine* , [sérialisé](#) , au *résultat* .
5. Si [le port](#) d' *origine* n'est pas nul, ajoutez un caractère U+003A COLON (:) et le [port](#) d' *origine* , [sérialisé](#) , au *résultat* .
6. Retourner *le résultat* .

La [sérialisation](#) de (" `https`", " `xn--maraa-rta.example`", null, null) est " `https://xn--maraa-rta.example`".

*Il y avait aussi une sérialisation Unicode d'une origine . Cependant, il n'a jamais été largement adopté.*

---

Deux [origines](#) , *A* et *B* , sont dites **identiques** si l'algorithme suivant renvoie true :

1. Si *A* et *B* sont la même [origine opaque](#) , alors retourne true.
2. Si *A* et *B* sont tous deux [des origines de tuple](#) et que leurs [schémas](#) , [hôtes](#) et [port](#) sont identiques, alors renvoie true.
3. Renvoie faux.

Deux [origines](#) , *A* et *B* , sont dites de **même domaine d'origine** si l'algorithme suivant renvoie true :

1. Si *A* et *B* sont la même [origine opaque](#) , alors retourne true.
2. Si *A* et *B* sont tous deux [des origines de tuple](#) , exécutez ces sous-étapes :

1. Si les schémas de *A* et *B* sont identiques et que leurs domaines sont identiques et non nuls, alors renvoie vrai.
2. Sinon, si *A* et *B* sont de même origine et que leurs domaines sont identiques et nuls, alors renvoie vrai.
3. Renvoie faux.

<i>UN</i>	<i>B</i>	<u>même origine</u>	<u>même domaine d'origine</u>
("https", "example.org", nul, nul)	("https", "example.org", nul, nul)	✓	✓
("https", "example.org", 314, nul)	("https", "example.org", 420, nul)	✗	✗
("https", "example.org", 314, "example.org")	("https", "example.org", 420, "example.org")	✗	✓
("https", "example.org", nul, nul)	("https", "example.org", nul, "example.org")	✓	✗
("https", "example.org", nul, "example.org")	("http", "example.org", nul, "example.org")	✗	✗

### 7.1.1.1 Sites

Un **schéma-et-hôte** est un tuple d'un **schéma** (une chaîne ASCII) et d'un **hôte** (un hôte).

Un **site** est une origine opaque ou un schéma-et-hôte.

Pour **obtenir un site**, étant donné une *origine* *origin*, exécutez ces étapes :

1. Si *origin* est une origine opaque, alors retourne *origin*.
2. Si le domaine enregistrable de l' hôte de l' *origine* est nul, alors retourne ( schéma de l'*origine*, hôte de l' *origine* ).
3. Retour ( schéma d' *origine*, domaine enregistrable de l' hôte d' *origine* ).

Deux sites, *A* et *B*, sont dits être **le même site** si l'algorithme suivant renvoie true :

1. Si *A* et *B* sont la même origine opaque, le retour est vrai.
2. Si *A* ou *B* est une origine opaque, alors retourne false.
3. Si les valeurs de schéma de *A* et *B* sont différentes, alors renvoie false.

4. Si les valeurs d'hôte de  $A$  et de  $B$  ne sont pas égales , alors renvoie false.
5. Renvoie vrai.

La **sérialisation d'un site** est la chaîne obtenue en appliquant l'algorithme suivant au site site donné :

1. Si le site est une origine opaque , alors retournez " `null`".
2. Soit le résultat le site [0].
3. Ajouter " `://`" au résultat .
4. Ajouter le site [1], sérialisé , au résultat .
5. Retourner le résultat .

*Il doit être clair à partir du contexte que la valeur sérialisée est un site, pas une origine, car il n'y a pas nécessairement de différence syntaxique entre les deux. Par exemple, l'origine (" `https`", " `shop.example`", `null`, `null`) et le site (" `https`", " `shop.example`") ont la même sérialisation : " `https://shop.example`".*

Deux origines ,  $A$  et  $B$  , sont dites être **sans schéma même site** si l'algorithme suivant renvoie vrai :

1. Si  $A$  et  $B$  sont la même origine opaque , alors retourne true.
2. Si  $A$  et  $B$  sont tous les deux des origines de tuple , alors :
  1. Soit  $hostA$  l' hôte de  $A$  , et  $hostB$  soit l' hôte de  $B$ .
  2. Si  $hostA$  est égal à  $hostB$  et que le domaine enregistrable de  $hostA$  est `null`, alors renvoie true.
  3. Si le domaine enregistrable de  $hostA$  est égal au domaine enregistrable de  $hostB$  et n'est pas nul, alors renvoie true.
3. Renvoie faux.

Deux origines ,  $A$  et  $B$  , sont dites être **le même site** si l'algorithme suivant renvoie true :

1. Soit  $siteA$  le résultat de l'obtention d'un site donné  $A$  .
2. Soit  $siteB$  le résultat de l'obtention d'un site donné  $B$  .
3. Si  $siteA$  est le même site que  $siteB$  , alors retourne true.
4. Renvoie faux.

Contrairement aux concepts [de même origine](#) et [de même origine-domaine](#) , pour [le même site](#) et [le même site sans schéma](#) , les composants [de port](#) et [de domaine](#) sont ignorés.

Pour les raisons [expliquées dans URL](#) , les concepts [de même site](#) et [sans schéma de site](#) doivent être évités dans la mesure du possible, au profit de [contrôles de même origine](#) .

Étant donné que `wildlife.museum`, `museum` et `com` sont [des suffixes publics](#) et que ce `example.com` n'est pas :

UN	B	<a href="#">même site sans schéma</a>	<a href="#">même site</a>
(" https", " example.com")	(" https", " sub.example.com")	✓	✓
(" https", " example.com")	(" https", " sub.other.example.com")	✓	✓
(" https", " example.com")	(" http", " non-secure.example.com")	✓	✗
(" https", " r.wildlife.museum")	(" https", " sub.r.wildlife.museum")	✓	✓
(" https", " r.wildlife.museum")	(" https", " sub.other.r.wildlife.museum")	✓	✓
(" https", " r.wildlife.museum")	(" https", " other.wildlife.museum")	✗	✗
(" https", " r.wildlife.museum")	(" https", " wildlife.museum")	✗	✗
(" https", " wildlife.museum")	(" https", " wildlife.museum")	✓	✓
(" https", " example.com")	(" https", " example.com.")	✗	✗

(Ici, nous avons omis les composants [de port](#) et [de domaine](#) car ils ne sont pas pris en compte.)

### 7.1.1.2 Assouplissement de la restriction de même origine

```
document.domain [ = domain ]
```

Renvoie le domaine actuel utilisé pour les contrôles de sécurité.

Peut être défini sur une valeur qui supprime les sous-domaines, pour modifier le [domaine](#) de l' [origine](#) afin de permettre aux pages d'autres sous-domaines du même domaine (si elles font la même chose) d'accéder les unes aux autres. Cela permet aux pages sur différents hôtes d'un domaine d'accéder de manière synchrone aux DOM de l'autre.

Dans `iframe` les `s` en bac à sable, `Document` les `s` avec [des origines opaques](#) et `Document` les `s` sans [contexte de navigation](#) , le setter lèvera

une exception "[SecurityError](#)". Dans les cas où [crossOriginIsolated](#) ou [originAgentCluster](#) retourne true, le setter ne fera rien.

Évitez d'utiliser le [document.domain](#) setter. Cela sape les protections de sécurité fournies par la politique de même origine. Ceci est particulièrement aigu lors de l'utilisation d'un hébergement mutualisé ; par exemple, si un tiers non approuvé est capable d'héberger un serveur HTTP à la même adresse IP mais sur un port différent, alors la protection de même origine qui protège normalement deux sites différents sur le même hôte échouera, car les ports sont ignorés lors de la comparaison des origines après [document.domain](#) l'utilisation du passeur.

En raison de ces failles de sécurité, cette fonctionnalité est en cours de suppression de la plateforme Web. (Il s'agit d'un long processus qui prend de nombreuses années.)

Au lieu de cela, utilisez [postMessage\(\)](#) ou [MessageChannel](#) des objets pour communiquer entre les origines de manière sûre.

Les [domain](#) étapes du getter sont :

1. Soit *effectiveDomain* le [domaine](#) effectif de [cette origine](#) .
2. Si *effectiveDomain* est null, renvoie la chaîne vide.
3. Renvoie *effectiveDomain* , [sérialisé](#) .

Les [domain](#) étapes du setter sont :

1. Si [ce contexte de navigation](#) est nul, lancez un "[SecurityError](#)" [DOMException](#) .
2. Si [cet indicateur de sandboxing actif](#) a son [indicateur de contexte de navigation](#) en [sandbox défini](#) [document.domain](#) , lancez un "[SecurityError](#)" [DOMException](#) .
3. Soit *effectiveDomain* le [domaine](#) effectif de [cette origine](#) .
4. Si *effectiveDomain* est null, lancez un "[SecurityError](#)" [DOMException](#) .
5. Si la valeur donnée [n'est pas un suffixe de domaine enregistrable et n'est pas égale à effectiveDomain](#) , lancez un "[SecurityError](#)" [DOMException](#) .
6. Si le [cluster d'agents](#) de l' [agent environnant](#) est [défini sur l'origine](#) est vrai, alors retournez.
7. Définissez [le domaine](#) de cette *origine* [sur](#) le résultat de [l'analyse](#) de la valeur donnée.

Pour déterminer si une [chaîne de valeur scalaire](#) *hostSuffixString* est un **suffixe de domaine enregistrable** ou est égal à un [hôte](#) *originalHost* :

1. Si *hostSuffixString* est la chaîne vide, renvoie false.
2. Soit *hostSuffix* le résultat de [l'analyse](#) de *hostSuffixString*.
3. Si *hostSuffix* est un échec, renvoie false.
4. Si *hostSuffix* n'est pas [égal à](#) *originalHost*, alors :
  1. Si *hostSuffix* ou *originalHost* n'est pas un [domaine](#), alors renvoyez false.

Cela exclut [les hôtes](#) qui sont [des adresses IP](#).

2. Si *hostSuffix*, préfixé par U+002E (.), ne correspond pas à la fin de *originalHost*, renvoie false.
3. Si l'un des éléments suivants est vrai
  - *hostSuffix* [est égal au suffixe public](#) de *hostSuffix*
  - *hostSuffix*, préfixé par U+002E (.), correspond à la fin du [suffixe public](#) *originalHost*

puis renvoie faux. [\[URL\]](#)

4. [Assert](#) : [le suffixe public](#) de *originalHost*, préfixé par U+002E (.), correspond à la fin de *hostSuffix*.

5. Renvoie vrai.

<i>hostSuffixString</i>	<i>hôte d'origine</i>	Le résultat de <a href="#">est un suffixe de domaine enregistrable de ou est égal à</a>	Remarques
"0.0.0.0"	0.0.0.0	✓	
"0x10203"	0.1.2.3	✓	
"[0::1]"	::1	✓	
"example.com"	example.com	✓	
"example.com"	example.com.	✗	Le point final est significatif.
"example.com."	example.com	✗	
"example.com"	www.example.com	✓	

" com"	example.com	✗	Au moment de la rédaction, com est un suffixe public.
" example"	example	✓	
" compute.amazonaws.com"	example.compute.amazonaws.com	✗	Au moment de la rédaction, *.compute.amazonaws.com est un suffixe public.
" example.compute.amazonaws.com"	www.example.compute.amazonaws.com	✗	
" amazonaws.com"	www.example.compute.amazonaws.com	✗	
" amazonaws.com"	test.amazonaws.com	✓	Au moment de la rédaction, amazonaws.com est un domaine enregistrable.

### 7.1.2 Clusters d'agents à clé d'origine

#### window.originAgentCluster

Renvoie true si this [Window](#) appartient à un [cluster d'agents](#) qui est [origin - keyed](#) , de la manière décrite dans cette section.

Un [Document](#) livré sur un [contexte sécurisé](#) peut demander qu'il soit placé dans un [cluster d'agents](#) à [clé d'origine](#) , en utilisant l' en-tête de réponse HTTP ``. Cet en-tête est un [en-tête structuré](#) dont la valeur doit être un [booléen](#) . [\[CHAMPS STRUCTURÉS\]](#) [Origin-Agent-Cluster](#)

Selon le modèle de traitement dans la [création et l'initialisation d'un nouvel Document objet](#) , les valeurs qui ne sont pas la valeur vraie [booléenne de l'en-tête structuré](#) (c'est-à-dire `` ?1`) seront ignorées.

Les conséquences de l'utilisation de cet en-tête sont que la [clé de cluster d'agents](#) [Document](#) résultante est son [origine](#) , au lieu du [site correspondant](#) . En termes d'effets observables, cela signifie que tenter d' [assouplir la restriction de même origine](#) en utilisant ne fera rien et qu'il ne sera pas possible d'envoyer des objets à des origines croisées (même s'ils sont [de même site](#) ). Dans les coulisses, cet isolement peut permettre aux agents utilisateurs d'allouer plus efficacement des ressources spécifiques à l'implémentation correspondant à [des clusters d'agents](#) , [tels que des processus ou des threads](#). [document.domainWebAssembly.ModuleDocument](#)

Notez qu'au sein d'un [groupe de contexte de navigation](#) , l'en-tête `` ne peut jamais entraîner l'arrivée d'objets [Origin-Agent-Cluster](#) de même origine dans différents [clusters d'agents](#) , même si l'un envoie l'en-tête et l'autre non. Ceci est empêché au moyen de la [carte de clé de cluster d'agent historique](#) . [Document](#)



Cela signifie que le `originAgentCluster` getter peut renvoyer `false`, même si l'en-tête est défini, si l'en-tête a été omis sur une page de même origine précédemment chargée dans le même [groupe de contexte de navigation](#) . De même, il peut renvoyer `true` même lorsque l'en-tête n'est pas défini.

Les `originAgentCluster` étapes getter consistent à renvoyer le cluster [d'agents](#) de [l'agent environnant avec la clé d'origine](#) .

[Document](#) les `s` avec une [origine opaque](#) peuvent être considérés comme inconditionnellement liés à l'origine ; pour eux, l'en-tête n'a aucun effet et le `originAgentCluster` getter renverra toujours `true`.

De la même manière, [Document](#) les `s` dont [le mode d'isolement cross-origin](#) du [cluster d'agents](#) n'est pas " " sont automatiquement définis sur l'origine. L'en-tête `` peut être utile comme indice supplémentaire pour les implémentations concernant l'allocation des ressources, puisque les en-têtes `` et `` utilisés pour réaliser l'isolation entre les origines visent davantage à garantir que tout ce qui se trouve dans le même espace d'adressage opte pour être là. Mais l'ajouter n'aurait aucun effet observable supplémentaire sur le code de l'auteur. [noneOrigin-Agent-ClusterCross-Origin-Opener-PolicyCross-Origin-Embedder-Policy](#)

### 7.1.3 Politiques d'ouverture d'origine croisée

Une **valeur de règle d'ouverture d'origine croisée** permet à un document auquel on accède dans un [contexte de navigation de niveau supérieur](#) de forcer la création d'un nouveau [contexte de navigation de niveau supérieur](#) et d'un [groupe](#) correspondant . Les valeurs possibles sont :

" **unsafe-none** "

Il s'agit de la valeur par défaut (actuelle) et signifie que le document occupera le même [contexte de navigation de niveau supérieur](#) que son prédécesseur, à moins que ce document ne spécifie une [stratégie d'ouverture croisée](#) différente .

" **same-origin-allow-popups** "

Cela force la création d'un nouveau [contexte de navigation de niveau supérieur](#) pour le document, à moins que son prédécesseur n'ait spécifié la même [politique d'ouverture cross-origin](#) et qu'ils soient [de même origine](#) .

" **same-origin** "

Cela se comporte de la même manière que " [same-origin-allow-popups](#) ", avec en plus que tout [contexte de navigation auxiliaire](#) créé doit contenir des documents [de même origine](#) qui ont également la même [politique d'ouverture d'origine croisée](#) ou il apparaîtra fermé à l'ouvreur.

## " same-origin-plus-COEP "

Cela se comporte de la même manière que " [same-origin](#)", avec en plus qu'il définit le [mode d'isolation cross-origin](#) du (nouveau) [groupe](#) du [contexte de navigation de niveau supérieur](#) sur " " ou " ".[logicalconcrete](#)

*" [same-origin-plus-COEP](#) " ne peut pas être défini directement via l' [Cross-Origin-Opener-Policy](#) en-tête `` , mais résulte d'une combinaison de la définition de `` et d'un en-tête `` dont la valeur est [compatible avec l'isolation d'origine croisée](#) ensemble. [Cross-Origin-Opener-Policy](#): [same-origin](#)[Cross-Origin-Embedder-Policy](#)*

Une **politique d'ouverture d'origine croisée** consiste en :

- Une **valeur** , qui est une [valeur de règle d'ouverture cross-origin](#) , initialement "[unsafe-none](#)".
- Un **point de terminaison de rapport** , qui est une chaîne ou null, initialement null.
- Une **valeur de rapport uniquement** , qui est une [valeur de règle d'ouverture cross-origin](#) , initialement "[unsafe-none](#)".
- Un **point de terminaison de rapport uniquement** , qui est une chaîne ou null, initialement null.

Pour **faire correspondre les valeurs de politique d'ouverture d'origine croisée** , étant donné une [valeur de politique d'ouverture d'origine croisée](#) *A* , une [origine](#) *originA* , une [valeur de politique d'ouverture d'origine croisée](#) *B* et une [origine](#) *originB* :

1. Si *A* est "[unsafe-none](#)" et *B* est "[unsafe-none](#)", alors retourne vrai.
2. Si *A* est "[unsafe-none](#)" ou *B* est "[unsafe-none](#)", alors retourne faux.
3. Si *A* est *B* et *originA* est [la même origine](#) que *originB* , alors retourne true.
4. Renvoie faux.

### 7.1.3.1 Les en-têtes



La [politique d'ouverture cross-origin](#)[Document](#) d' est dérivée des en-têtes de réponse HTTP `` et ``. Ces en-têtes sont [des en-têtes structurés](#) dont la valeur doit être un [jeton](#) . [\[CHAMPS STRUCTURÉS\]](#)[Cross-Origin-Opener-Policy](#)[Cross-Origin-Opener-Policy-Report-Only](#)

Les valeurs [de jeton](#) valides sont les [valeurs de stratégie d'ouverture](#) . [Le jeton peut également avoir des paramètres](#) attachés ; parmi ceux-ci, le **report-to** paramètre " " peut avoir une [chaîne d'URL valide](#) identifiant un point de terminaison de rapport approprié. [\[SIGNALLEMENT\]](#)

*Selon le modèle de traitement décrit ci-dessous, les agents utilisateurs ignoreront cet en-tête s'il contient une valeur invalide. De même, les agents utilisateurs ignoreront cet en-tête si la valeur ne peut pas être analysée comme un [jeton](#) .*

---

Pour **obtenir une politique d'ouverture cross-origin** donnée une [réponse](#) et un [environnement](#) *reservedEnvironment* :

1. Soit *policy* une nouvelle [politique d'ouverture cross-origin](#) .
2. Si *reservedEnvironment* est un [contexte non sécurisé](#) , alors retourne *policy* .
3. Soit *parsedItem* le résultat de [l'obtention d'une valeur de champ structuré](#) donnée par ``Cross-Origin-Opener-Policy`` et " *item* " à partir de [la liste d'en-tête](#) de [la réponse non sécurisée](#) de *la réponse* .
4. Si *parsedItem* n'est pas nul, alors :
  1. Si *parsedItem* [0] est " [same-origin](#) ", alors :
    1. Soit *coop* le résultat de [l'obtention d'une politique d'intégration d'origines croisées](#) à partir de *response* et *reservedEnvironment* .
    2. Si [la valeur](#) de *coop* est [compatible avec l'isolation cross-origin](#) , alors définissez [la valeur](#) de *policy* sur " [.same-origin-plus-COEP](#) " .
    3. Sinon, définissez [la valeur](#) de *la stratégie* sur " [.same-origin](#) " .
  2. Si *parsedItem* [0] est " [same-origin-allow-popups](#) ", alors définissez [la valeur](#) de *policy* sur " [.same-origin-allow-popups](#) " .
  3. Si *parsedItem* [1][ " [report-to](#) " ] [existe](#) et qu'il s'agit d'une chaîne, définissez [le point de terminaison de rapport](#) de *la stratégie* sur *parsedItem* [1][ " [report-to](#) " ] .
5. Définissez *parsedItem* sur le résultat de [l'obtention d'une valeur de champ structuré](#) donnée par ``Cross-Origin-Opener-Policy-Report-Only`` et " *item* " à partir de [la liste d'en-tête](#) de *la réponse* .
6. Si *parsedItem* n'est pas nul, alors :

1. Si `parsedItem [0]` est "same-origin", alors :

1. Soit `coop` le résultat de l'obtention d'une politique d'intégration d'origines croisées à partir de `response` et `reservedEnvironment`.
2. Si la valeur de `coop` est compatible avec l'isolement cross-origin ou si la valeur report-only de `coop` est compatible avec l'isolement cross-origin, alors définissez la valeur report-only de `policy` sur ".same-origin-plus-COEP".

*Rapport uniquement COOP prend également en compte le rapport COEP uniquement pour attribuer la same-origin-plus-COEP valeur spéciale ". Cela permet aux développeurs plus de liberté dans l'ordre de déploiement de COOP et COEP.*

3. Sinon, définissez la valeur de rapport uniquement de la stratégie sur ".same-origin".
2. Si `parsedItem [0]` est "same-origin-allow-popups", alors définissez la valeur de rapport uniquement de la stratégie sur ".same-origin-allow-popups".
3. Si `parsedItem [1][report-to]` existe et qu'il s'agit d'une chaîne, définissez le point de terminaison de rapport uniquement de la stratégie sur `parsedItem [1][report-to]`.

7. Politique de retour .

### 7.1.3.2 Changements de groupe de contexte de navigation en raison de la politique d'ouverture d'origine croisée

Pour **vérifier si les valeurs COOP nécessitent un changement de groupe de contexte de navigation**, étant donné un booléen `isInitialAboutBlank`, deux origines `responseOrigin` et `activeDocumentNavigationOrigin`, et deux valeurs de stratégie d'ouverture cross-origin `responseCOOPValue` et `activeDocumentCOOPValue` :

1. Si le résultat de la correspondance `activeDocumentCOOPValue`, `activeDocumentNavigationOrigin`, `responseCOOPValue` et `responseOrigin` est true, renvoie false.
2. Si toutes les conditions suivantes sont vraies :
  - `isInitialAboutBlank` ;
  - La valeur de `activeDocumentCOOPValue` est "same-origin-allow-popups" ; et

- La valeur de réponse COOP est "[unsafe-none](#)"

puis renvoie faux.

3. Renvoie vrai.

Pour **vérifier si l'application de COOP uniquement pour les rapports nécessiterait un changement de groupe de contexte de navigation**, étant donné un booléen `isInitialAboutBlank`, deux [origines](#) `responseOrigin`, `activeDocumentNavigationOrigin` et deux [politiques d'ouverture inter-origines](#) `responseCOOP` et `activeDocumentCOOP` :

1. Si le résultat de [la vérification si les valeurs COOP nécessitent un changement de groupe de contexte de navigation](#) étant donné `isInitialAboutBlank`, `responseOrigin`, `activeDocumentNavigationOrigin`, [la valeur de rapport uniquement de](#) `responseCOOP` et la [valeur de rapport uniquement](#) de `activeDocumentCOOPReportOnly` est false, alors renvoie false.

*La correspondance des politiques de rapport uniquement permet à un site Web de spécifier la même politique d'ouverture d'origine croisée de rapport uniquement sur toutes ses pages et de ne pas recevoir de rapports de violation pour les navigations entre ces pages.*

2. Si le résultat de [la vérification si les valeurs COOP nécessitent un changement de groupe de contexte de navigation](#) étant donné `isInitialAboutBlank`, `responseOrigin`, `activeDocumentNavigationOrigin`, [la valeur](#) de `responseCOOP` et [la valeur de rapport uniquement](#) de `activeDocumentCOOPReportOnly` est true, alors renvoie true.
3. Si le résultat de [la vérification si les valeurs COOP nécessitent un changement de groupe de contexte de navigation](#) étant donné `isInitialAboutBlank`, `responseOrigin`, `activeDocumentNavigationOrigin`, [la valeur de rapport uniquement de](#) `responseCOOP` et la [valeur](#) de `activeDocumentCOOPReportOnly` est true, alors renvoie true.

4. Renvoie faux.

Un **résultat d'application de la politique d'ouverture d'origine croisée** est une [structure](#) avec les [éléments](#) suivants :

- Un booléen **a besoin d'un groupe de contexte de navigation switch**, initialement false.
- Un booléen **aurait besoin d'un changement de groupe de contexte de navigation en raison de report-only**, initialement false.
- Une [URL](#) URL .
- Une [origine](#) origine .

- Une [politique d'ouverture cross-origin](#) **politique d'ouverture cross-origin** .
- Un **contexte booléen courant est navigation source** , initialement faux.

Pour **appliquer la politique d'ouverture d'origine croisée d'une réponse** , étant donné un [contexte de navigation scanningContext](#) , une [URL](#) *responseURL* , une [origine](#) *responseOrigin* , une [politique d'ouverture d'origine croisée](#) *responseCOOP* , un [résultat d'application de la politique d'ouverture d'origine croisée](#) *currentCOOPEnforcementResult* et un référent [referrer](#) :

1. Laissez *newCOOPEnforcementResult* être un nouveau [résultat d'application de la politique d'ouverture d'origine croisée](#) avec  
  
[nécessite un changement de groupe de contexte de navigation](#)  
*currentCOOPEnforcementResult* a [besoin d'un changement de groupe de contexte de navigation](#)  
[aurait besoin d'un changement de groupe de contexte de navigation en raison du rapport uniquement](#)  
*currentCOOPEnforcementResult* aurait [besoin d'un changement de groupe de contexte de navigation en raison du rapport uniquement](#)  
[URL](#)  
*URL de réponse*  
[origine](#)  
*réponseOrigine*  
[politique d'ouverture d'origine croisée](#)  
*réponseCOOP*  
[le contexte actuel est la source de navigation](#)  
vrai
2. Soit *isInitialAboutBlank* le document [actif](#) du *contexte de navigation* [is initial](#) [.about:blank](#)
3. Si *isInitialAboutBlank* est true et [que l'URL initiale](#) de *scanningContext* est nulle, définissez [l'URL initiale](#) de *scanningContext* sur *responseURL* .
4. Si le résultat de [la vérification si les valeurs COOP nécessitent un changement de groupe de contexte de navigation](#) donné *isInitialAboutBlank* , [la valeur](#) de la [politique d'ouverture cross-origin](#) de *currentCOOPEnforcementResult* , [l'origine](#) de *currentCOOPEnforcementResult* , [la valeur](#) de *responseCOOP* et *responseOrigin* est vrai, alors :
  1. Définissez *newCOOPEnforcementResult* 's [a besoin d'un commutateur de groupe de contexte de navigation](#) sur true.
  2. Si [la taille](#) de l'ensemble de [contextes de navigation](#) du [groupe](#) *scanningContext* est supérieure à 1, alors :
    1. [Mettre en file d'attente un rapport de violation pour le changement de groupe de contexte de navigation lors de la navigation vers une réponse COOP](#) avec *responseCOOP* , " *enforce* " , *responseURL* , l' [url](#)



de `currentCOOPEnforcementResult` ,  
l' origine de `currentCOOPEnforcementResult` , `responseOrigin` et  
`referrer` .

2. Mettez en file d'attente un rapport de violation pour le changement de groupe de contexte de navigation lorsque vous vous éloignez d'une réponse COOP avec la politique d'ouverture croisée de `currentCOOPEnforcementResult` ,  
" ", l'url de `currentCOOPEnforcementResult` , `responseURL` , l'origine de `currentCOOPEnforcementResult` , `responseOrigin` et le contexte actuel de `currentCOOPEnforcementResult` est source de navigation .`enforce`
5. Si le résultat de la vérification si l'application de COOP uniquement pour les rapports nécessiterait un changement de groupe de contexte de navigation donné `isInitialAboutBlank` , `responseOrigin` , `currentCOOPEnforcementResult` 's origine , `responseCOOP` et `currentCOOPEnforcementResult` 's cross-origin opener policy , est vrai, alors :
  1. Définir le résultat nécessiterait un changement de groupe de contexte de navigation en raison du rapport uniquement sur `true`.
  2. Si la taille de l'ensemble de contextes de navigation du groupe `scanningContext` est supérieure à 1, alors :
    1. Mettre en file d'attente un rapport de violation pour le changement de groupe de contexte de navigation lors de la navigation vers une réponse COOP avec `responseCOOP` ,  
" `reporting` ", `responseURL` , l' url  
de `currentCOOPEnforcementResult` ,  
l' origine de `currentCOOPEnforcementResult` , `responseOrigin` et  
`referrer` .
    2. Mettez en file d'attente un rapport de violation pour le changement de groupe de contexte de navigation lorsque vous vous éloignez d'une réponse COOP avec la politique d'ouverture croisée de `currentCOOPEnforcementResult` ,  
" ", l'url de `currentCOOPEnforcementResult` , `responseURL` , l'origine de `currentCOOPEnforcementResult` , `responseOrigin` et le contexte actuel de `currentCOOPEnforcementResult` est source de navigation .`reporting`
6. Renvoie `newCOOPEnforcementResult` .

Pour **obtenir un contexte de navigation à utiliser pour une réponse de navigation** , étant donné un contexte de navigation `scanningContext` , un ensemble d'indicateurs de sandboxing `sandboxFlags` , une politique d'ouverture d'origine croisée `navigationCOOP` et un résultat d'application de politique d'ouverture d'origine croisée `coopEnforcementResult` :

1. Si *browserContext* n'est pas un [contexte de navigation de niveau supérieur](#) , alors retourne *scanningContext* .
2. Si *coopEnforcementResult* a [besoin d'un changement de groupe de contexte de navigation](#) est faux, alors :
  1. Si *coopEnforcementResult* aurait [besoin d'un changement de groupe de contexte de navigation en raison de](#) la valeur de rapport uniquement, définissez [l'ID de groupe de contexte de navigation virtuel](#) du contexte de navigation sur un nouvel identifiant unique.
  2. Retourne *scanningContext* .
3. Laissez *newBrowsingContext* être la première valeur de retour de [la création d'un nouveau contexte de navigation de niveau supérieur et document](#) .

*Dans ce cas, nous allons effectuer un échange de groupe de contexte de navigation. browserContext ne sera pas utilisé par le new Document que nous sommes sur le point de créer . S'il n'est pas non Document plus utilisé par d'autres s (comme ceux du cache arrière/avant), l'agent utilisateur peut le détruire à ce stade.*

4. Si la [valeur](#) de *navigationCOOP* est " " , alors définissez [le mode d'isolation cross-origin](#) du [groupe](#) *newBrowsingContext* sur " " ou " " . Le choix est [défini par l'implémentation](#) [.same-origin-plus-COEPlogicalconcrete](#)

*Il est difficile sur certaines plates-formes de fournir les propriétés de sécurité requises par la [capacité isolée cross-origin](#) . " [concrete](#) " autorise l'accès et " [logical](#) " ne le permet pas.*

5. Si *sandboxFlags* n'est pas vide, alors :
  1. Affirmer que [la valeur](#) de *navigationCOOP* est " " . [.unsafe-none](#)
  2. [Assert](#) : [le jeu d'indicateurs de sandboxing](#) contextuel de *newBrowsingContext* [est vide](#) .
  3. Définissez [l'indicateur de sandboxing](#) contextuel de *newBrowsingContext* sur un [clone](#) de *sandboxFlags* .
6. Renvoie *newBrowsingContext* .

### 7.1.3.3 Rapports

Une **relation accessible par un accesseur** est une énumération qui décrit la relation entre deux [contextes de navigation](#) entre lesquels un accès s'est produit. Il peut prendre les valeurs suivantes :



### ***l'accesseur est l'ouvreur***

Le [contexte de navigation](#) de l'accesseur ou l'un de ses [ancêtres](#) est le [contexte de navigation d'ouverture](#) du [contexte de navigation de niveau supérieur](#) du contexte de navigation accédé .

### ***l'accesseur est ouvert***

Le [contexte de navigation](#) accédé ou l'un de ses [ancêtres](#) est le [contexte de navigation d'ouverture](#) du [contexte de navigation de niveau supérieur du contexte de navigation de](#) l'accesseur .

### ***aucun***

Il n'y a pas de relation d'ouverture entre le [contexte de navigation](#) de l'accesseur , le [contexte de navigation](#) de l'accesseur ou l'un de leurs [ancêtres](#) .

Pour **vérifier si un accès entre deux contextes de navigation doit être signalé** , étant donné deux [contextes de navigation](#) *accesseur* et *accédé* , un nom de propriété JavaScript *P* , et un [objet de paramètres d'environnement](#) *environment* :

1. Si *P* n'est pas un [nom de propriété de fenêtre accessible d'origine croisée](#) , alors retournez.
2. **Assert** : le [document actif](#) de l'*accesseur* et le [document actif](#) de l'*accédé* sont tous deux [entièrement actifs](#) .
3. Soit *accessorTopDocument* le [document actif](#) du [contexte de navigation de niveau supérieur](#) de l'*accesseur* .
4. Soit *accessorInclusiveAncestorOrigins* la liste obtenue en prenant l'[origine](#) du [document actif](#) de chacun des navigables [ancêtres inclusifs](#) du [document actif de l'accesseur](#) .
5. Laisser *accessedTopDocument* accéder au [document actif](#) du contexte [de navigation de niveau supérieur](#) .
6. Soit *accessedInclusiveAncestorOrigins* la liste obtenue en prenant l'[origine](#) du [document actif](#) de chacun des [navigables ancêtres inclusifs](#) du document [actif accédé](#) .
7. Si l'un des *accessorInclusiveAncestorOrigins* n'a pas [la même origine](#) que l'[origine](#) de *accessorTopDocument* , ou si l'un des *accessedInclusiveAncestorOrigins* n'a pas [la même origine](#) que l'[origine](#) de l'*accessorTopDocument* , alors retournez.

*Cela évite la fuite d'informations sur les iframes d'origine croisée vers un cadre de niveau supérieur avec des rapports de politique d'ouverture d'origine croisée.*

8. Si l'[ID de groupe de contextes de navigation virtuels](#) du contexte de navigation [de niveau supérieur](#) de l'*accesseur* est accédé à l'[ID de groupe de](#)

[contextes de navigation virtuels](#) du contexte de navigation [de niveau supérieur](#) , alors retour.

9. Soit *accessorAccessedRelationship* une nouvelle [relation accessible par l'accesseur](#) avec la valeur [none](#) .
10. Si le contexte de navigation d'ouverture du [contexte de navigation de niveau supérieur de l'](#) accès est *accessor* ou est un [ancêtre](#) de *accessor* , alors définissez *accessorAccessedRelationship* sur [accessor is opener](#) .
11. Si le contexte de navigation d'ouverture du [contexte de navigation de niveau supérieur](#) de l' *accesseur* est *accédé* ou est un [ancêtre](#) de *accessor* , alors définissez *accessorAccessedRelationship* sur [accessor is openee](#) .
12. [Rapports de violation de file d'attente pour les accès](#) , étant donné *accessorAccessedRelationship* , [politique d'ouverture cross-origin](#) d' *accessorTopDocument* , [politique d'ouverture cross-origin](#) d' *accessorTopDocument* , URL du [document actif de l' accesseur](#) , URL du [document actif](#) de l' *accesseur* , *top- URL initiale* du [contexte de navigation de niveau](#) , accessible URL *initiale* du [contexte de navigation de niveau supérieur](#) , *accesseurorigine* du [document actif](#) de l' accès , origine du [document actif de l' accès](#) , *origine* de l' ouverture du contexte [de navigation de niveau supérieur de l' accesseur](#) à la création , *origine* de l' ouverture du [contexte de navigation de niveau supérieur de l' accès à la création](#) , *accesseurTopDocument* 's [renvoyer](#) , a *accédé* au [renvoyer](#) , *P* et à l' environnement de *TopDocument* .

Pour **nettoyer une URL à envoyer dans un rapport** donné une URL [URL](#) :

1. Soit *sanitizedURL* une copie de *url* .
2. [Définissez le nom d'utilisateur](#) donné *sanitizedURL* et la chaîne vide.
3. [Définissez le mot de passe](#) indiqué par *sanitizedURL* et la chaîne vide.
4. Renvoie la [sérialisation](#) de *sanitizedURL* avec [le fragment d'exclusion](#) défini sur true.

Pour **mettre en file d'attente un rapport de violation pour le changement de groupe de contexte de navigation lors de la navigation vers une réponse COOP** donnée une [politique d'ouverture d'origine croisée](#) *coop* , une *disposition* de chaîne , une [URL](#) *coopURL* , une [URL](#) *previousResponseURL* , deux [origines](#) *coopOrigin* et *previousResponseOrigin* , et un référent [renvoyer](#) :

1. Si [le point de terminaison de rapport](#) de *coop* est nul, retournez.
2. Soit *coopValue* la [valeur](#) de *coop* .
3. Si *disposition* est " `reporting` ", alors définissez *coopValue* sur [la valeur de rapport uniquement](#) de *coop* .

4. Soit *serializedReferrer* une chaîne vide.
5. Si *referrer* est une [URL](#) , définissez *serializedReferrer* sur la [sérialisation](#) de *referrer* .
6. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	<i>disposition</i>
effectivePolicy	<i>coopValeur</i>
URL de réponse précédente	Si <i>coopOrigin</i> et <i>previousResponseOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>previousResponseURL</i> , null sinon.
réfèrent	<i>serializedReferrer</i>
taper	" navigation-to-response "

7. Corps [de la file d'attente](#) en tant que " *coop* " pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* .

Pour **mettre en file d'attente un rapport de violation pour le changement de groupe de contexte de navigation lorsque vous vous éloignez d'une réponse COOP** en fonction d'une [politique d'ouverture d'origine croisée](#) *coop* , d'une *disposition* de chaîne , d'une [URL](#) *coopURL* , d'une [URL](#) *nextResponseURL* , de deux [origines](#) *coopOrigin* et *nextResponseOrigin* , et d'un booléen *isCOOPResponseNavigationSource* :

1. Si [le point de terminaison de rapport](#) de *coop* est nul, retournez.
2. Soit *coopValue* la [valeur](#) de *coop* .
3. Si *disposition* est " *reporting* ", alors définissez *coopValue* sur [la valeur de rapport uniquement](#) de *coop* .
4. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	<i>disposition</i>
effectivePolicy	<i>coopValeur</i>
URL de réponse suivante	Si <i>coopOrigin</i> et <i>nextResponseOrigin</i> ont <a href="#">la même origine</a> ou si <i>isCOOPResponseNavigationSource</i> est vrai, il s'agit du <a href="#">nettoyage</a> de <i>previousResponseURL</i> , null sinon.
taper	" navigation-from-response "

5. Corps [de la file d'attente](#) en tant que " *coop* " pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* .

Pour mettre en file d'attente les rapports **de violation pour les accès** , étant donné une [relation accédée par l'accesseur](#) *accessorAccessedRelationship* , deux [règles](#)

d'ouverture d'origine croisée *accessorCOOP* et *accessedCOOP* ,  
quatre URL *accessorURL* , *accessorURL* , *accessorInitialURL* , *accessedInitialURL* ,  
quatre origines *accessorOrigin* , *accessorOrigin* , *accessorCreatorOrigin* et *accessed  
CreatorOrigin* , deux référénts *accessorReferrer* et *accessorReferrer* , un  
string *propertyName* , et un objet de paramètres d' *environnement* *environment* :

1. Si le point de terminaison de rapport de *coop* est nul, retournez.
2. Soit *coopValue* la valeur de *coop* .
3. Si *disposition* est " *reporting*", alors définissez *coopValue* sur la valeur de  
rapport uniquement de *coop* .
4. Si *accessorAccessedRelationship* est accessor is opener :
  1. Mettre en file d'attente un rapport de violation pour l'accès à une fenêtre  
ouverte , étant  
donné *accessorCOOP* , *accessorURL* , *accessedURL* , *accessedInitial  
URL* , *accessorOrigin* , *accessOrigin* , *accessCreatorOrigin* , *propertyName*  
et *environment* .
  2. Mettre en file d'attente un rapport de violation pour l'accès à partir de  
l'ouvreur , en fonction de *l'accessedCOOP* , de *l'accessorURL* , de  
*l'accessorURL* , de *l'accessorOrigin* , de *l'accessorOrigin* , du  
*propertyName* et de *l'accessedReferrer* .
5. Sinon, si *accessorAccessedRelationship* est accessor is openee :
  1. Mettre en file d'attente un rapport de violation pour l'accès à l'ouvreur ,  
étant  
donné *accessorCOOP* , *accessorURL* , *accessorURL* , *accessorOrigin* ,  
*accessOrigin* , *propertyName* , *accessorReferrer* et *environment* .
  2. Mettre en file d'attente un rapport de violation pour l'accès à partir d'une  
fenêtre ouverte , en fonction de *l'accessedCOOP* , de  
*l'accessorURL* , de *l'accessorURL* , de *l'accessorInitialURL* , de  
*l'accessorOrigin* , de *l'accessorOrigin* , de *l'accessorCreatorOrigin* et du  
*propertyName* .
6. Sinon:
  1. Mettre en file d'attente un rapport de violation pour accéder à une autre  
fenêtre , étant  
donné *accessorCOOP* , *accessorURL* , *accessorURL* , *accessorOrigin* ,  
*accessOrigin* , *propertyName* et *environnement*
  2. Mettez en file d'attente un rapport de violation pour l'accès à partir  
d'une autre fenêtre , en fonction de *l'accessedCOOP* , de  
*l'accessorURL* , de *l'accessorURL* , de *l'accessorOrigin* , de  
*l'accessorOrigin* et du *propertyName* .

Pour **mettre en file d'attente un rapport de violation pour l'accès à l'opener** , étant donné une [règle d'ouverture cross-origin](#) *coop* , deux [URL](#) *coopURL* et *openerURL* , deux [origines](#) *coopOrigin* et *openerOrigin* , une chaîne *propertyName* , un référent [referrer](#) et un objet [de paramètres d'environnement](#) *environment* :

1. Laissez *sourceFile* , *lineNumber* et *columnNumber* être l'URL de script pertinente et la position problématique qui ont déclenché ce rapport.
2. Soit *serializedReferrer* une chaîne vide.
3. Si *referrer* est une [URL](#) , définissez *serializedReferrer* sur la [sérialisation](#) de *referrer* .
4. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	" reporting"
effectivePolicy	<a href="#">valeur de rapport uniquement</a> de <i>coop</i>
propriété	<i>nom de la propriété</i>
URL d'ouverture	Si <i>coopOrigin</i> et <i>openerOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>openerURL</i> , null sinon.
réfèrent	<i>serializedReferrer</i>
fichier source	<i>fichier source</i>
numéro de ligne	<i>numéro de ligne</i>
numéro de colonne	<i>numéro de colonne</i>
taper	" access-to-opener"

5. Corps [de la file d'attente](#) en tant que " *coop*" pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* et *environment* .

Pour **mettre en file d'attente un rapport de violation pour l'accès à une fenêtre ouverte** , étant donné une [politique d'ouverture d'origine croisée](#) *coop* , trois [URL](#) *coopURL* , *openWindowURL* et *initialWindowURL* , trois [origines](#) *coopOrigin* , *openWindowOrigin* et *openerInitialOrigin* , une chaîne *propertyName* et un [objet de paramètres d'environnement](#) *environment* :

1. Laissez *sourceFile* , *lineNumber* et *columnNumber* être l'URL de script pertinente et la position problématique qui ont déclenché ce rapport.
2. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	" reporting"
effectivePolicy	<a href="#">valeur de rapport uniquement</a> de <i>coop</i>

clé	valeur
propriété	<i>nom de la propriété</i>
openWindowURL	Si <i>coopOrigin</i> et <i>openWindowOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>openWindowURL</i> , null sinon.
openWindowInitialURL	Si <i>coopOrigin</i> et <i>openerInitialOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>initialWindowURL</i> , null sinon.
fichier source	<i>fichier source</i>
numéro de ligne	<i>numéro de ligne</i>
numéro de colonne	<i>numéro de colonne</i>
taper	" access-to-opener"

3. Corps [de la file d'attente](#) en tant que " *coop*" pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* et *environnement* .

Pour **mettre en file d'attente un rapport de violation pour accéder à une autre fenêtre** , étant donné une [règle d'ouverture cross-origin](#) *coop* , deux [URL](#) *coopURL* et *otherURL* , deux [origines](#) *coopOrigin* et *otherOrigin* , une chaîne *propertyName* et un [objet de paramètres d'environnement](#) *environment* :

1. Laissez *sourceFile* , *lineNumber* et *columnNumber* être l'URL de script pertinente et la position problématique qui ont déclenché ce rapport.
2. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	" reporting"
effectivePolicy	<a href="#">valeur de rapport uniquement</a> de <i>coop</i>
propriété	<i>nom de la propriété</i>
autreURL	Si <i>coopOrigin</i> et <i>otherOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>otherURL</i> , null sinon.
fichier source	<i>fichier source</i>
numéro de ligne	<i>numéro de ligne</i>
numéro de colonne	<i>numéro de colonne</i>
taper	" access-to-opener"

3. Corps [de la file d'attente](#) en tant que " *coop*" pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* et *environnement* .

Pour **mettre en file d'attente un rapport de violation pour l'accès à partir de l'opener** , étant donné une [règle d'ouverture cross-origin](#) *coop* , deux [URL](#) *coopURL* et *openerURL* , deux [origines](#) *coopOrigin* et *openerOrigin* , une chaîne *propertyName* et un référent [referrer](#) :

1. Si [le point de terminaison de rapport](#) de *coop* est nul, retournez.
2. Soit *serializedReferrer* une chaîne vide.
3. Si *referrer* est une [URL](#) , définissez *serializedReferrer* sur la [sérialisation](#) de *referrer* .
4. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	" reporting"
effectivePolicy	<a href="#">valeur de rapport uniquement</a> de <i>coop</i>
propriété	<i>nom de la propriété</i>
URL d'ouverture	Si <i>coopOrigin</i> et <i>openerOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>openerURL</i> , null sinon.
réfèrent	<i>serializedReferrer</i>
taper	" access-to-opener"

5. Corps [de la file d'attente](#) en tant que " *coop*" pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* .

Pour **mettre en file d'attente un rapport de violation pour l'accès à partir d'une fenêtre ouverte** , étant donné une [politique d'ouverture d'origine croisée](#) *coop* , trois [URL](#) *coopURL* , *openWindowURL* et *initialWindowURL* , trois [origines](#) *coopOrigin* , *openWindowOrigin* et *openerInitialOrigin* , et une chaîne *propertyName* :

1. Si [le point de terminaison de rapport](#) de *coop* est nul, retournez.
2. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	" reporting"
effectivePolicy	<i>coopValeur</i>
propriété	<a href="#">valeur de rapport uniquement</a> de <i>coop</i>
openWindowURL	Si <i>coopOrigin</i> et <i>openWindowOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>openWindowURL</i> , null sinon.
openWindowInitialURL	Si <i>coopOrigin</i> et <i>openerInitialOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>initialWindowURL</i> , null sinon.
taper	" access-to-opener"

3. Corps [de la file d'attente](#) en tant que " *coop*" pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* .

Pour **mettre en file d'attente un rapport de violation pour l'accès à partir d'une autre fenêtre** , étant donné une [politique d'ouverture d'origine croisée](#) *coop* ,



deux [URL](#) *coopURL* et *otherURL* , deux [origines](#) *coopOrigin* et *otherOrigin* et une chaîne *propertyName* :

1. Si [le point de terminaison de rapport](#) de *coop* est nul, retournez.
2. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
disposition	" reporting"
effectivePolicy	<a href="#">valeur de rapport uniquement</a> de <i>coop</i>
propriété	<i>nom de la propriété</i>
autreURL	Si <i>coopOrigin</i> et <i>otherOrigin</i> ont <a href="#">la même origine</a> , il s'agit du <a href="#">nettoyage</a> de <i>otherURL</i> , null sinon.
taper	access-to-opener

3. Corps [de la file d'attente](#) en tant que " *coop*" pour [le point de terminaison de rapport](#) de *coop* avec *coopURL* .

## 7.1.4 Politiques d'intégration inter-origines



Une **valeur de stratégie d'intégration** est l'une des trois chaînes qui contrôlent la récupération de ressources d'origine croisée sans l'autorisation explicite des propriétaires de ressources.

### " **unsafe-none** "

Ceci est la valeur par défaut. Lorsque cette valeur est utilisée, les ressources cross-origine peuvent être récupérées sans donner d'autorisation explicite via le [protocole CORS](#) ou l' [Cross-Origin-Resource-Policy](#) en-tête ``.

### " **require-corp** "

Lorsque cette valeur est utilisée, la récupération des ressources d'origine croisée nécessite l'autorisation explicite du serveur via le [protocole CORS](#) ou l' [Cross-Origin-Resource-Policy](#) en-tête ``.

### " **credentialless** "

Lorsque cette valeur est utilisée, la récupération de ressources cross-origine no-CORS omet les informations d'identification. En échange, un [Cross-Origin-Resource-Policy](#) en-tête `` explicite n'est pas requis. D'autres requêtes envoyées avec des informations d'identification nécessitent l'autorisation explicite du serveur via le [protocole CORS](#) ou l' [Cross-Origin-Resource-Policy](#) en-tête ``.



**Avant de prendre en charge "credentialless", les implémenteurs sont fortement encouragés à prendre en charge à la fois :**

- Accès au réseau privé
- Blocage de réponse opaque

**Sinon, cela permettrait aux attaquants de tirer parti de la position réseau du client pour lire des ressources non publiques, en utilisant la capacité d'isolement d'origine croisée .**

Une valeur de politique d'intégration est compatible avec l'isolation d'origine croisée si elle est "credentialless" ou "require-corp".

Une **politique d'intégration** consiste en :

- Une **valeur** , qui est une valeur de politique d' intégration , initialement "unsafe-none".
- Une chaîne **de point de terminaison de rapport** , initialement la chaîne vide.
- Une **valeur de rapport uniquement** , qui est une valeur de politique d'intégration , initialement "unsafe-none".
- Un **rapport ne signalant que** la chaîne de point de terminaison, initialement la chaîne vide.

Le "**coop**" **type de rapport** est un type de rapport dont la valeur est "**coop**". Il est visible à ReportingObserverl'art .

#### 7.1.4.1 Les en-têtes

Les en-têtes de réponse HTTP ` **Cross-Origin-Embedder-Policy** ` et ` **Cross-Origin-Embedder-Policy-Report-Only** ` permettent à un serveur de déclarer une politique d'intégration pour un objet de paramètres d'environnement . Ces en-têtes sont des en-têtes structurés dont les valeurs doivent être token . [CHAMPS STRUCTURÉS]

Les valeurs de jeton valides sont les valeurs de stratégie d'intégration . Le jeton peut également avoir des paramètres attachés ; parmi ceux-ci, le **report-to** paramètre " " peut avoir une chaîne d'URL valide identifiant un point de terminaison de rapport approprié. [SIGNALEMENT]

*Le modèle de traitement échoue à s'ouvrir (par défaut à "unsafe-none") en présence d'un en-tête qui ne peut pas être analysé comme un jeton. Cela inclut les listes créées par inadvertance en combinant plusieurs instances de l'Cross-Origin-Embedder-Policy en-tête ` ` présent dans une réponse donnée :*

<u>Cross-Origin-Embedder-Policy</u>	<u>Valeur</u> finale de la stratégie d'intégration
Aucun en-tête livré	" <u>unsafe-none</u> "
`require-corp`	" <u>require-corp</u> "
`unknown-value`	" <u>unsafe-none</u> "
`require-corp, unknown-value`	" <u>unsafe-none</u> "
`unknown-value, unknown-value`	" <u>unsafe-none</u> "
`unknown-value, require-corp`	" <u>unsafe-none</u> "
`require-corp, require-corp`	" <u>unsafe-none</u> "

(Il en va de même pour Cross-Origin-Embedder-Policy-Report-Only.)

Pour **obtenir une politique d'incorporation** à partir d'une *réponse* response et d'un *environnement* environment :

1. Soit *policy* une nouvelle politique d'incorporation .
2. Si *environment* est un contexte non sécurisé , alors retourne *policy* .
3. Soit *parsedItem* le résultat de l'obtention d'une valeur de champ structuré avec Cross-Origin-Embedder-Policy et " *item*" à partir de la liste d'en-tête de la *réponse* .
4. Si *parsedItem* est non null et *parsedItem* [0] est compatible avec l'isolation cross-origin :
  1. Définissez la valeur de la *stratégie* sur *parsedItem* [0].
  2. Si *parsedItem* [1][report-to] existe , définissez le point de terminaison de la *stratégie* sur *parsedItem* [1][report-to]
5. Définissez *parsedItem* sur le résultat de l'obtention d'une valeur de champ structuré avec Cross-Origin-Embedder-Policy-Report-Only et " *item*" à partir de la liste d'en-tête de la *réponse* .
6. Si *parsedItem* est non null et *parsedItem* [0] est compatible avec l'isolation cross-origin :
  1. Définissez la valeur de rapport uniquement de la *stratégie* sur *parsedItem* [0].
  2. Si *parsedItem* [1][report-to] existe , définissez le point de terminaison de la *stratégie* sur *parsedItem* [1][report-to]

## 7. Politique de retour .

### 7.1.4.2 Contrôles de politique d'incorporation

Pour **vérifier la conformité d'une réponse de navigation à sa politique d'intégration** étant donné une *réponse* [response](#) , un *navigable* [navigable](#) et une [politique d'intégration](#) *responsePolicy* :

1. Si *navigable* n'est pas un [enfant navigable](#) , alors retourne true.
2. Soit *parentPolicy* la politique d' [incorporation](#) du conteneur de la [politique](#) du [document](#) *navigable* .
3. Si [la valeur de rapport uniquement](#) de *parentPolicy* est [compatible avec l'isolation d'origine croisée](#) et que [la valeur](#) de *responsePolicy* ne l'est pas, mettez [en file d'attente une violation d'héritage de stratégie d'intégration d'origine croisée](#) avec la *réponse* , " " , [le point de terminaison de rapport de parentPolicy uniquement](#) , " " , et l'objet de [paramètres pertinent](#) du [document conteneur](#) de *navigable* .`navigationreporting`
4. Si [la valeur](#) de *parentPolicy* n'est pas [compatible avec l'isolation cross-origin](#) ou si [la valeur](#) de *responsePolicy* est [compatible avec l'isolation cross-origin](#) , alors retourne true.
5. [Mettre en file d'attente une violation d'héritage de stratégie d'intégration d'origines croisées](#) avec la *réponse* , " `navigation` " , [le point de terminaison de rapport de parentPolicy](#) , " " et l' [objet de paramètres pertinent](#) du [document conteneur](#) de *navigable* .`enforce`
6. Renvoie faux.

Pour **vérifier la stratégie d'intégration d'un objet global** en fonction d'un [WorkerGlobalScope](#) *workerGlobalScope* , d'un *propriétaire* [d'objet de paramètres d'environnement](#) et d' une [réponse](#) :

1. Si *workerGlobalScope* n'est pas un [DedicatedWorkerGlobalScope](#) objet, renvoie true.
2. Soit *policy* la [politique d'intégration](#) de *workerGlobalScope* .
3. Soit *ownerPolicy* la [politique d'incorporation](#) du conteneur de [politique](#) du *propriétaire* .
4. Si [la valeur de rapport uniquement](#) de *ownerPolicy* est [compatible avec l'isolation cross-origin](#) et que [la valeur](#) de *policy* ne l'est pas, [mettez en file d'attente une violation d'héritage de politique d'intégration cross-origin](#) avec la *réponse* , " " , [le point de terminaison de rapport uniquement](#) de la *politique* du *propriétaire* , " " , et *propriétaire* .`worker initializationreporting`

5. Si [la valeur](#) de *ownerPolicy* n'est pas [compatible avec l'isolation cross-origin](#) ou si [la valeur](#) de *policy* est [compatible avec l'isolation cross-origin](#) , alors retourne true.
6. [Mettre en file d'attente une violation d'héritage de stratégie d'intégration d'origines croisées](#) avec la réponse , " `worker initialization`" , [le point de terminaison de rapport](#) de la *stratégie du propriétaire* , " " et le *propriétaire* .`enforce`
7. Renvoi faux.

Pour **mettre en file d'attente une violation d'héritage de politique d'intégration cross-origin** en fonction d'une réponse [response](#) , d'un type de chaîne , d'un point de terminaison de chaîne , d'une disposition de chaîne et d'un objet de paramètres [d'environnement](#) :

1. Soit *sérialisé* le résultat de [la sérialisation d'une URL de réponse pour les rapports](#) avec *response* .
2. Soit *body* un nouvel objet contenant les propriétés suivantes :

clé	valeur
taper	<i>taper</i>
URL bloquée	<i>sérialisé</i>
disposition	<i>disposition</i>

3. *Corps de la file d'attente* en tant que "`coop`" [type de rapport](#) pour le point de terminaison sur les paramètres .

### 7.1.5 Bac à sable

Un **ensemble d'indicateurs de sandboxing** est un ensemble de zéro ou plusieurs des indicateurs suivants, qui sont utilisés pour restreindre les capacités dont disposent les ressources potentiellement non approuvées :

#### L' *indicateur de contexte de navigation de navigation en bac à sable*

Cet indicateur [empêche le contenu de naviguer dans des contextes de navigation autres que le contexte de navigation en bac à sable lui-même](#) (ou les contextes de navigation imbriqués à l'intérieur de celui-ci), [les contextes de navigation auxiliaires](#) (qui sont protégés par l' [indicateur de contexte de navigation de navigation auxiliaire en bac à sable](#) défini ci-après) et la [navigation de niveau supérieur. contextuel](#) (qui est protégé par l' [indicateur de navigation de niveau supérieur en bac à sable sans activation de l'utilisateur](#) et l'indicateur de contexte de navigation [de niveau supérieur en bac à sable avec l'indicateur de contexte de navigation d'activation de l'utilisateur](#) définis ci-dessous).

Si l' [indicateur de contexte de navigation de navigation auxiliaire en bac à sable](#) n'est pas défini, dans certains cas, les restrictions autorisent néanmoins l'ouverture de popups (nouveaux [contextes de navigation de niveau supérieur](#) ). Ces [contextes de navigation](#) ont toujours **un navigateur en bac à sable autorisé** , défini lors de la création du contexte de navigation, qui permet au [contexte de navigation](#) qui les a créés de les parcourir. (Sinon, l' [indicateur de contexte de navigation de navigation en bac à sable](#) les empêcherait de naviguer même s'ils étaient ouverts.)

#### **L' *indicateur de contexte de navigation de navigation auxiliaire en bac à sable***

Ce drapeau [empêche le contenu de créer de nouveaux contextes de navigation auxiliaires](#) , par exemple en utilisant l' `target` attribut ou la `window.open()` méthode.

#### **L' *indicateur de contexte de navigation de niveau supérieur en bac à sable sans activation de l'utilisateur***

Cet indicateur [empêche le contenu de naviguer dans son contexte de navigation de niveau supérieur](#) et [empêche le contenu de fermer son contexte de navigation de niveau supérieur](#) . Elle est consultée uniquement lorsque la [fenêtre active](#) du contexte de navigation sandbox n'a pas [d'activation transitoire](#) .

Lorsque l' [indicateur de contexte de navigation de navigation de niveau supérieur en bac à sable sans activation de l'utilisateur](#) n'est pas défini, le contenu peut naviguer dans son [contexte de navigation de niveau supérieur](#) , mais d'autres [contextes de navigation](#) sont toujours protégés par l' [indicateur de contexte de navigation de navigation en bac à sable](#) et éventuellement l' [indicateur de contexte de navigation de navigation auxiliaire en bac à sable](#) .

#### **La *navigation de niveau supérieur en bac à sable avec l'indicateur de contexte de navigation d'activation de l'utilisateur***

Cet indicateur [empêche le contenu de naviguer dans son contexte de navigation de niveau supérieur](#) et [empêche le contenu de fermer son contexte de navigation de niveau supérieur](#) . Il n'est consulté que lorsque la [fenêtre active](#) du contexte de navigation en bac à sable a [une activation transitoire](#) .

Comme avec l' [indicateur de contexte de navigation de niveau supérieur en bac à sable sans activation de l'utilisateur](#) , cet indicateur n'affecte que le [contexte de navigation de niveau supérieur](#) ; s'il n'est pas défini, d'autres [contextes de navigation](#) peuvent toujours être protégés par d'autres drapeaux.

#### **L' *indicateur de contexte de navigation d'origine en bac à sable***

Cet indicateur [force le contenu dans une origine unique](#) , l'empêchant ainsi d'accéder à d'autres contenus de la même [origine](#) .

Cet indicateur [empêche également le script de lire ou d'écrire dans l' `document.cookie` attribut IDL](#) et bloque l'accès à [localStorage](#).

### **L' indicateur de contexte de navigation des formulaires en bac à sable**

Cet drapeau [bloque la soumission du formulaire](#) .

### **L' indicateur de contexte de navigation de verrouillage du pointeur en bac à sable**

Cet indicateur désactive l'API Pointer Lock. [\[POINTERLOCK\]](#)

### **L' indicateur de contexte de navigation des scripts en bac à sable**

Cet indicateur [bloque l'exécution du script](#) .

### **L' indicateur de contexte de navigation des fonctionnalités automatiques en bac à sable**

Cet indicateur bloque les fonctionnalités qui se déclenchent automatiquement, telles que [la lecture automatique d'une vidéo](#) ou [la mise au point automatique d'un contrôle de formulaire](#) .

### **L' indicateur de contexte de navigation en bac à sable**`document.domain`

Cet indicateur empêche le contenu d'utiliser le `document.domain`setter.

### **Le bac à sable se propage à l'indicateur de contextes de navigation auxiliaires**

Cet indicateur empêche le contenu de s'échapper du bac à sable en garantissant que tout [contexte de navigation auxiliaire](#) qu'il crée hérite de l' [ensemble d'indicateurs de bac à sable actif](#) du contenu .

### **Le drapeau des modaux en bac à sable**

Cet indicateur empêche le contenu d'utiliser l'une des fonctionnalités suivantes pour produire des boîtes de dialogue modales :

- `window.alert()`
- `window.confirm()`
- `window.print()`
- `window.prompt()`
- l' `beforeunload` événement

### **L' indicateur de contexte de navigation de verrouillage d'orientation en bac à sable**

Cet indicateur désactive la possibilité de verrouiller l'orientation de l'écran. [\[ORIENTATION DE L'ÉCRAN\]](#)

### **L' indicateur de contexte de navigation de la présentation en bac à sable**

Cet indicateur désactive l'API de présentation. [\[PRÉSENTATION\]](#)

### **L' indicateur de contexte de navigation des téléchargements en bac à sable**

Cet indicateur empêche le contenu de lancer ou d'instancier des téléchargements, que ce soit via [le téléchargement d'hyperliens](#) ou via [la navigation](#) qui est traitée [comme un téléchargement](#) .

### **L' indicateur de contexte de navigation de navigation des protocoles personnalisés en bac à sable**

Cet indicateur empêche les navigations vers [des schémas non récupérés](#) d'être [transmises à un logiciel externe](#) .

Lorsque l'agent utilisateur doit **analyser une directive de sandboxing** , étant donné une *entrée* de chaîne , une *sortie* [d'ensemble d'indicateurs de sandboxing](#) , il doit exécuter les étapes suivantes :

1. [Fractionner l'entrée sur les espaces blancs ASCII](#) , pour obtenir *des jetons* .
2. Laissez *la sortie* être vide.
3. Ajoutez les drapeaux suivants à *la sortie* :
  - L' [indicateur de contexte de navigation de navigation en bac à sable](#) .
  - L' [indicateur de contexte de navigation de navigation auxiliaire en bac à sable](#) , à moins que *jetons* ne contienne le **allow-popups** mot-clé.
  - L' [indicateur de contexte de navigation de niveau supérieur en sandbox sans activation de l'utilisateur](#) , à moins que *jetons* ne contienne le **allow-top-navigation** mot-clé.
  - La [navigation de niveau supérieur en bac à sable avec l'indicateur de contexte de navigation d'activation de l'utilisateur](#) , à moins que *les jetons* ne contiennent le **allow-top-navigation-by-user-activation** mot-clé ou le [allow-top-navigation](#) mot-clé.

*Cela signifie que si le [allow-top-navigation](#) est présent, le [allow-top-navigation-by-user-activation](#) mot-clé n'aura aucun effet. Pour cette raison, spécifier les deux est une erreur de conformité de document.*

- L' [indicateur de contexte de navigation d'origine en bac à sable](#) , sauf si les *jetons* contiennent le **allow-same-origin** mot-clé.

*Le [allow-same-origin](#) mot clé est destiné à deux cas.*

*Tout d'abord, il peut être utilisé pour autoriser le contenu du même site à être mis en bac à sable pour désactiver les scripts, tout en permettant l'accès au DOM du contenu en bac à sable.*

*Deuxièmement, il peut être utilisé pour intégrer le contenu d'un site tiers, mis en bac à sable pour empêcher ce site d'ouvrir des fenêtres contextuelles, etc., sans empêcher la page intégrée de communiquer*



avec son site d'origine, en utilisant les API de base de données pour stocker des données, etc.

- L' [indicateur de contexte de navigation des formulaires en bac à sable](#) , à moins que *jetons* ne contienne le `allow-forms` mot-clé.
- L' [indicateur de contexte de navigation de verrouillage du pointeur en bac à sable](#) , à moins que *jetons* ne contienne le `allow-pointer-lock` mot-clé.
- L' [indicateur de contexte de navigation des scripts en bac à sable](#) , à moins que *jetons* ne contienne le `allow-scripts` mot-clé.
- L' [indicateur de contexte de navigation des fonctionnalités automatiques en bac à sable](#) , à moins que *les jetons* ne contiennent le `allow-scripts` mot-clé (défini ci-dessus).

*Cet indicateur est assoupli par le même mot-clé que les scripts, car lorsque les scripts sont activés, ces fonctionnalités sont de toute façon trivialement possibles, et il serait regrettable de forcer les auteurs à utiliser des scripts pour les faire lorsqu'ils sont en bac à sable plutôt que de leur permettre d'utiliser les fonctionnalités déclaratives.*

- L' [indicateur de contexte de navigation en bac à sable](#) `document.domain` .
  - Le [bac à sable se propage aux contextes de navigation auxiliaires flag](#) , sauf si *tokens* contient le `allow-popups-to-escape-sandbox` mot-clé.
  - L' [indicateur modal en bac à sable](#) , sauf si *jetons* contient le `allow-modals` mot-clé.
  - L' [indicateur de contexte de navigation de verrouillage d'orientation en bac à sable](#) , à moins que *jetons* ne contienne le `allow-orientation-lock` mot-clé.
  - L' [indicateur de contexte de navigation de la présentation en bac à sable](#) , à moins que *jetons* ne contienne le `allow-presentation` mot-clé.
  - L' [indicateur de contexte de navigation des téléchargements en bac à sable](#) , à moins que *jetons* ne contienne le `allow-downloads` mot-clé.
  - L' [indicateur de contexte de navigation de navigation des protocoles personnalisés en bac à sable](#) , sauf si *les jetons* contiennent le `allow-top-navigation-to-custom-protocols` mot clé, le `allow-popups` mot clé ou le `allow-top-navigation` mot clé.
-



Chaque [contexte de navigation de niveau supérieur](#) possède un **ensemble d'indicateurs de sandboxing contextuel**, qui est un [ensemble d'indicateurs de sandboxing](#). Lorsqu'un [contexte de navigation](#) est créé, son [ensemble d'indicateurs de sandboxing contextuel](#) doit être vide. Il est renseigné par [les règles permettant de choisir un navigable](#) et d' [obtenir un contexte de navigation à utiliser pour un](#) algorithme de réponse de navigation.

Chaque [iframe](#) élément a un **iframe ensemble d'indicateurs de sandboxing**, qui est un [ensemble d'indicateurs de sandboxing](#). Les indicateurs d'un [iframe ensemble d'indicateurs de sandboxing](#) qui sont définis à un moment donné sont déterminés par l'attribut [iframe](#) de l'élément [.sandbox](#)

Chaque [Document](#) a un **ensemble d'indicateurs de sandboxing actif**, qui est un [ensemble d'indicateurs de sandboxing](#). Lorsque le [Document](#) est créé, son [ensemble d'indicateurs de sandboxing actif](#) doit être vide. Il est rempli par l' [algorithme de navigation](#).

Chaque [liste CSP](#) *cspList* a **des indicateurs de sandboxing dérivés de CSP**, qui est un [ensemble d'indicateurs de sandboxing](#). C'est la valeur de retour de l'algorithme suivant :

1. Soit *directives* un [ensemble ordonné vide](#).
2. [Pour chaque](#) stratégie dans *cspList* :
  1. Si [la disposition](#) de la *politique* n'est pas " ", alors [continuez](#) *.enforce*
  2. Si [le jeu de directives](#) de la stratégie [contient](#) une [directive](#) dont le nom est " ", alors [ajoutez](#) cette directive à *directives* [. sandbox](#)
3. Si *directives* est vide, renvoie un [jeu d'indicateurs de sandboxing](#) vide.
4. Soit *directive* des *directives* [ [taille](#) des *directives* – 1].
5. Renvoie le résultat de l'analyse de la *directive* [sandboxing directive](#).

---

Pour **déterminer les drapeaux de sandboxing de création** pour un *contexte de navigation* [context de navigation](#), étant donné null ou un *embedder* d'élément, renvoyez l' [union](#) des drapeaux qui sont présents dans les [ensembles de drapeaux de sandboxing](#) suivants :

- Si *embedder* est null, alors : les indicateurs définis sur [l'indicateur de sandboxing](#) contextuel du *contexte de navigation* sont définis.

- Si *embedder* est un élément, alors : les indicateurs définis sur l'indicateur de sandboxing de l' *intégrateur* [iframe](#) [sont définis](#) .
- Si l'*intégrateur* est un élément, alors : les indicateurs définis sur le nœud de l'*intégrateur* [document](#) l' [indicateur de bac à sable actif défini](#) .

### 7.1.6 Conteneurs de politique

Un **conteneur de stratégie** est une [structure](#) contenant des stratégies qui s'appliquent à un [Document](#), un [WorkerGlobalScope](#) ou un [WorkletGlobalScope](#). Il comporte les [éléments](#) suivants :

- Une **liste CSP** , qui est une [liste CSP](#) . Il est initialement vide.
- Une **stratégie d'intégration** , qui est une [stratégie d'intégration](#) . Il s'agit initialement d'une nouvelle [politique d'intégration](#) .
- Une **stratégie de référence** , qui est une [stratégie de référence](#) . Il s'agit initialement de la [stratégie de référence par défaut](#) .

Déplacez d'autres stratégies dans le conteneur de stratégies.

Pour **cloner un conteneur de stratégie** à partir d'un [conteneur de stratégie](#) *policyContainer* :

1. Soit *clone* un nouveau [conteneur de stratégie](#) .
2. [Pour chaque](#) *stratégie* dans [la liste CSP](#) de *policyContainer* , [ajoutez](#) une copie de *la stratégie* dans [la liste CSP](#) du *clone* .
3. Définissez [la stratégie d'intégration](#) du *clone* sur une copie de [la stratégie d'intégration](#) de *policyContainer* .
4. Définissez [la politique de référence](#) du *clone* sur [la politique de référence](#) de *policyContainer* .
5. *Clone* de retour .

Pour déterminer si une *URL* [d'URL](#) **nécessite de stocker le conteneur de stratégie dans l'historique** :

1. Si [le schéma](#) de l' *url* est " " , alors renvoie `false.blob`
2. Si *url* [est local](#) , alors retourne `true`.
3. Renvoie faux.

Pour créer un conteneur de **stratégie à partir d'une réponse de récupération** donnée une *réponse* [de réponse](#) et un [environnement](#) -ou-null :

1. Si [le schéma](#) de l' [URL](#) de la *réponse* est " ", renvoie alors un [clone](#) du [conteneur](#) de stratégie de l' [environnement de l'entrée](#) de l' [URL](#) blob de l'URL de la *réponse* `.blob`
2. Soit *result* un nouveau [conteneur de politique](#) .
3. Définissez [la liste CSP](#) de *result* sur le résultat de [l'analyse des politiques de sécurité du contenu d'](#) une réponse donnée *response* .
4. Si *environment* n'est pas nul, alors définissez [la politique d'incorporation](#) de *result* sur le résultat de [l'obtention d'une politique d'incorporation](#) donnée *response* et *environment* . Sinon, réglez-le sur " ".[unsafe-none](#)
5. Définissez [la politique de référence](#) du *resultat* sur le résultat de [l'analyse de l'en-tête ``](#) en réponse donnée . [\[POLITIQUE DE RÉFÉRENCE\]](#)[Referrer-Policy](#)
6. Retourner *le résultat* .

Pour **déterminer le conteneur de stratégie de paramètres de navigation** en fonction d'une [URL](#) *responseURL* et de quatre [conteneurs de stratégie](#) -ou-nulls *historyPolicyContainer* , *initiatorPolicyContainer* , *parentPolicyContainer* et *responsePolicyContainer* :

1. Si *historyPolicyContainer* n'est pas nul, alors :
  1. [Assert](#) : *responseURL* [nécessite le stockage du conteneur de stratégie dans l'historique](#) .
  2. Renvoie un [clone](#) de *historyPolicyContainer* .
2. Si *responseURL* est [about:srcdoc](#) , alors :
  1. [Assert](#) : *parentPolicyContainer* n'est pas nul.
  2. Renvoie un [clone](#) de *parentPolicyContainer* .
3. Si *responseURL* [est local](#) et que *initiatorPolicyContainer* n'est pas nul, renvoyez un [clone](#) de *initiatorPolicyContainer* .
4. Si *responsePolicyContainer* n'est pas null, retournez *responsePolicyContainer* .
5. Renvoie un nouveau [conteneur de stratégie](#) .

Pour **initialiser le conteneur de stratégie d'une étendue globale de travail** en fonction d'un [WorkerGlobalScope](#) *workerGlobalScope* , d'une réponse [response](#) et d'un environnement [environment](#) :

1. Si l'[URL](#) de *workerGlobalScope* [est locale](#) mais que son [schéma](#) n'est pas " " : blob
  1. [Assert](#) : [la taille](#) de [l'ensemble propriétaire](#) de *workerGlobalScope* est 1.
  2. Définissez [le conteneur de stratégie](#) de *workerGlobalScope* sur un [clone](#) du [conteneur de stratégie](#) de [l'objet de paramètres pertinent](#) [0] de l' ensemble [propriétaire de](#) *workerGlobalScope* .
2. Sinon, définissez [le conteneur de stratégie](#) de *workerGlobalScope* sur le résultat de [la création d'un conteneur de stratégie à partir d'une réponse d'extraction](#) en fonction de *la réponse* et de *l'environnement* .

## 7.2 API liées à la navigation et à l'historique des sessions

### 7.2.1 Infrastructure de sécurité pour les objets [Window](#), [WindowProxy](#) et [Location](#)

Bien qu'il soit généralement impossible d'accéder aux objets à travers [les origines](#) , la plate-forme Web ne serait pas fidèle à elle-même si elle ne disposait pas d'exceptions héritées à cette règle dont dépend le Web.

Cette section utilise la terminologie et les conventions typographiques de la spécification JavaScript. [\[JAVASCRIPT\]](#)

#### 7.2.1.1 Intégration avec IDL

Lorsqu'un [contrôle de sécurité](#) est appelé, avec un *platformObject* , *identifier* et *type* , exécutez ces étapes :

1. Si *platformObject* n'est pas un objet [Window](#) ou [Location](#), alors retournez.
2. Pour chaque *e* de [CrossOriginProperties](#) ( *platformObject* ) :
  1. Si [SameValue](#) ( *e* .[[Property]], *identifier* ) est vrai, alors :
    1. Si *le type* est " *method* " et que *e* n'a ni [[NeedsGet]] ni [[NeedsSet]], alors retournez.

2. Sinon, si *type* est " `getter`" et `e.[[NeedsGet]]` est vrai, alors retournez.
3. Sinon, si *type* est " `setter`" et `e.[[NeedsSet]]` est vrai, alors retournez.
3. Si [isPlatformObjectSameOrigin](#) ( *platformObject* ) est faux, lancez un `"_SecurityError_" DOMException` .

#### 7.2.1.2 Emplacement interne partagé : `[[CrossOriginPropertyDescriptorMap]]`

`Window` et `Location` les objets ont tous deux un emplacement interne `[[CrossOriginPropertyDescriptorMap]]` , dont la valeur est initialement une carte vide.

Le slot interne [\[\[CrossOriginPropertyDescriptorMap\]\]](#) contient une carte avec des entrées dont les clés sont ( *currentGlobal* , *objectGlobal* , *propertyKey* )-tuples et les valeurs sont des descripteurs de propriété, comme une mémorisation de ce qui est visible pour les scripts lorsque *currentGlobal* inspecte un objet `Window` ou `objectGlobal` . Il est rempli paresseusement par [CrossOriginGetOwnPropertyHelper](#) , qui le consulte lors de futures recherches. `Location`

Les agents utilisateurs doivent permettre à une valeur contenue dans la carte d'être ramassée avec sa clé correspondante lorsque rien ne contient de référence à une partie de la valeur. Autrement dit, tant que la collecte des ordures n'est pas observable.

Par exemple, `const href = Object.getOwnPropertyDescriptor(crossOriginLocation, "href").set` la valeur et sa clé correspondante dans la carte ne peuvent pas être ramassées car cela serait observable.

Les agents utilisateurs peuvent avoir une optimisation par laquelle ils suppriment les paires clé-valeur de la carte lorsque `document.domain` est défini. Ceci n'est pas observable car `document.domain` ne peut pas revenir sur une valeur antérieure.

Par exemple, définir `document.domain` sur " `example.com`" sur `www.example.com` signifie que les agents utilisateurs peuvent supprimer toutes les paires clé-valeur de la carte où une partie de la clé est `www.example.com`, car cela ne peut plus jamais faire partie de l' [origine](#) et donc la valeur correspondante n'a jamais pu être extraite de la carte.

#### 7.2.1.3 Opérations abstraites partagées

### 7.2.1.3.1 CrossOriginProperties ( O )

1. [Assert](#) : O est un [Location](#) ou [Window](#) un objet.
2. Si O est un [Location](#) objet, alors retournez « { [[Property]] : " href", [[NeedsGet]] : false, [[NeedsSet]] : true }, { [[Property]] : " " replace } ».
3. Renvoie « { [[Propriété]] : "window", [[NeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[Propriété]] : " " », { [[ selfNeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[Propriété]] : " location", [[NeedsGet]] : vrai, [[NeedsSet]] : vrai }, { [[Propriété]] : " close", { [[Propriété]] : " closed", [[NeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[Propriété]] : " focus", { [[Propriété]] : " blur", { [[Propriété]] : " frames", [[NeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[Propriété]] : " ", [[ lengthNeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[ Propriété]] : " top", [[NeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[Property]] : " ", [ openerNeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[Propriété]] : "parent", [[NeedsGet]] : vrai, [[NeedsSet]] : faux }, { [[Propriété]] : " postMessage" } ».

*Cette opération abstraite ne renvoie pas de [Completion Record](#) .  
Les propriétés indexées n'ont pas besoin d'être safelistées dans cet algorithme, car elles sont gérées directement par l' [WindowProxy](#) objet.*

Un nom de propriété JavaScript *P* est un **nom de propriété de fenêtre accessible cross-origin** s'il est " window", " self", " " location, " " close, " ", " closed", " " focus, " blur", " frames" length, " " top, " opener", " parent", " postMessage" ou un [nom de propriété d'index de tableau](#) .

### 7.2.1.3.2 CrossOriginPropertyFallback ( P )

1. Si *P* est " then", [@@toStringTag](#) , [@@hasInstance](#) ou [@@isConcatSpreadable](#) , alors renvoyez [PropertyDescriptor](#) { [[Value]] : non défini, [[Writable]] : false, [[Enumerable]] : false, [[Configurable]] : vrai }.
2. Lancez un [\\_SecurityError\\_ DOMException](#) .

### 7.2.1.3.3 IsPlatformObjectSameOrigin ( O )

1. Renvoie vrai si l' [origine](#) de [l'objet de paramètres actuel](#) est [le même domaine d'origine](#) que [l'origine](#) de [l'objet de paramètres pertinent](#) de O , et faux dans le cas contraire.

*Cette opération abstraite ne renvoie pas de [Completion Record](#) .*

Ici, l' objet de paramètres actuel correspond à peu près à "l'appelant", car cette vérification a lieu avant que le contexte d'exécution du getter/setter/méthode en question n'atteigne la pile de contexte d'exécution JavaScript . Par exemple, dans le code `w.document`, cette étape est invoquée avant que le `document` getter ne soit atteint dans le cadre de l'algorithme `[[Get]]` pour le `WindowProxy w` .

#### 7.2.1.3.4 CrossOriginGetOwnPropertyHelper ( O , P )

Si cette opération abstraite renvoie undefined et qu'il n'y a pas de comportement personnalisé, l'appelant doit lancer un `"SecurityError" DOMException` . En pratique, cela est géré par l'appelant appelant CrossOriginPropertyFallback .

1. Soit `crossOriginKey` un tuple composé de l' objet de paramètres actuel , de l' objet de paramètres pertinent de O et de P .
2. Pour chaque e de CrossOriginProperties ( O ):
  1. Si SameValue ( e.[[Property]], P ) est vrai, alors :
    1. Si la valeur de l' emplacement interne [[CrossOriginPropertyDescriptorMap]] de O contient une entrée dont la clé est `crossOriginKey` , alors renvoyez la valeur de cette entrée.
    2. Soit `originalDesc` être OrdinaryGetOwnProperty ( O , P ).
    3. Laissez `crossOriginDesc` être indéfini.
    4. Si e.[[NeedsGet]] et e.[[NeedsSet]] sont absents, alors :
      1. Soit `value` soit `originalDesc` .[[Value]].
      2. Si IsCallable ( `value` ) est true, alors définissez `value` sur une fonction intégrée anonyme, créée dans le domaine actuel , qui effectue les mêmes étapes que l'opération IDL P sur l'objet O .
      3. Définissez `crossOriginDesc` sur PropertyDescriptor {  
[[Value]] : `value` , [[Enumerable]] : false, [[Writable]] : false,  
[[Configurable]] : true }.
  5. Sinon:
    1. Laissez `crossOriginGet` être indéfini.
    2. Si e.[[NeedsGet]] est vrai, alors définissez `crossOriginGet` sur une fonction intégrée anonyme, créée dans le domaine actuel , qui effectue les



mêmes étapes que le getter de l'attribut IDL *P* sur l'objet *O* .

3. Laissez *crossOriginSet* être indéfini.
  4. Si *e* .*[[NeedsSet]]* est vrai, alors définissez *crossOriginSet* sur une fonction intégrée anonyme, créée dans le [domaine actuel](#) , qui effectue les mêmes étapes que le setter de l'attribut IDL *P* sur l'objet *O* .
  5. Définissez *crossOriginDesc* sur [PropertyDescriptor](#) {  
  *[[Get]]* : *crossOriginGet* , *[[Set]]* : *crossOriginSet* ,  
  *[[Enumerable]]* : false, *[[Configurable]]* : true }.
  6. Créez une entrée dans la valeur de l'emplacement interne [\[\[CrossOriginPropertyDescriptorMap\]\]](#) de *O* avec la clé *crossOriginKey* et la valeur *crossOriginDesc* .
  7. Renvoie *crossOriginDesc* .
3. Retour indéfini.

*Cette opération abstraite ne renvoie pas de [Completion Record](#) .*  
*La raison pour laquelle les descripteurs de propriétés produits ici sont configurables est de préserver les [invariants des méthodes internes essentielles](#) requises par la spécification JavaScript. En particulier, étant donné que la valeur de la propriété peut changer en conséquence de la navigation, il est nécessaire que la propriété soit configurable. (Cependant, voir [tc39/ecma262 issue #672](#) et ses références ailleurs dans cette spécification pour les cas où nous ne sommes pas en mesure de préserver ces invariants, pour la compatibilité avec le contenu Web existant.) [\[JAVASCRIPT\]](#)*  
*La raison pour laquelle les descripteurs de propriété ne sont pas énumérables, malgré cette incompatibilité avec le comportement de même origine, est pour la compatibilité avec le contenu Web existant. Voir [le numéro 3183](#) pour plus de détails.*

#### **7.2.1.3.5 CrossOriginGet ( *O* , *P* , Récepteur )**

1. Soit *desc* ? *O* .*[[GetOwnProperty]]*( *P* ).
2. [Assert](#) : *desc* n'est pas indéfini.
3. Si [IsDataDescriptor](#) ( *desc* ) est vrai, alors renvoie *desc* .*[[Value]]*.
4. [Assert](#) : [IsAccessorDescriptor](#) ( *desc* ) est vrai.
5. Soit *getter* *desc* .*[[Get ]]*.



6. Si *getter* n'est pas défini, lancez un `"_SecurityError"_DOMException`.
7. Retour ? [Appel](#) ( *getter* , *Receiver* ).

#### 7.2.1.3.6 *CrossOriginSet* ( *O* , *P* , *V* , Récepteur )

1. Soit *desc* ? *O* .[[GetOwnProperty]]( *P* ).
2. [Assert](#) : *desc* n'est pas indéfini.
3. Si *desc* .[[Set]] est présent et sa valeur n'est pas indéfinie, alors :
  1. Effectuer ? [Appel](#) ( *Passeur* , *Récepteur* , « *V* » ).
  2. Renvoie vrai.
4. Lancez un `"_SecurityError"_DOMException`.

#### 7.2.1.3.7 *CrossOriginOwnPropertyKeys* ( *O* )

1. Soit *keys* une nouvelle [List](#) vide .
2. Pour chaque *e* de [CrossOriginProperties](#) ( *O* ), [ajoutez](#) *e* .[[Property]] aux *keys* .
3. Renvoie la concaténation des *clés* et «  
" then", [@@toStringTag](#) , [@@hasInstance](#) , [@@isConcatSpreadable](#) ».

Cette opération abstraite ne renvoie pas de [Completion Record](#) .

### 7.2.2 L' [Window](#) objet



[Global=Window,

Exposed=Window,

[LegacyUnenumerableNamedProperties](#)]

```
interface Window : EventTarget {
```

```
    // the current browsing context
```

```
    [LegacyUnforgeable] readonly attribute WindowProxy window;
```

```
    [Replaceable] readonly attribute WindowProxy self;
```

```
    [LegacyUnforgeable] readonly attribute Document document;
```

```
    attribute DOMString name;
```

```
    [PutForwards=href, LegacyUnforgeable] readonly attribute
```

```
    Location location;
```

```
    readonly attribute History history;
```

```
    readonly attribute CustomElementRegistry customElements;
```

```
    [Replaceable] readonly attribute BarProp locationbar;
```

```
    [Replaceable] readonly attribute BarProp menubar;
```

```
    [Replaceable] readonly attribute BarProp personalbar;
```

```
    [Replaceable] readonly attribute BarProp scrollbars;
```

```
    [Replaceable] readonly attribute BarProp statusbar;
```

```
    [Replaceable] readonly attribute BarProp toolbar;
```

```
    attribute DOMString status;
```

```
    undefined close();
```

```
    readonly attribute boolean closed;
```

```
    undefined stop();
```

```
    undefined focus();
```

```
    undefined blur();
```

```
// other browsing contexts
```

```
[Replaceable] readonly attribute WindowProxy frames;
```

```
[Replaceable] readonly attribute unsigned long length;
```

```
[LegacyUnforgeable] readonly attribute WindowProxy? top;
```

```
attribute any opener;
```

```
[Replaceable] readonly attribute WindowProxy? parent;
```

```
readonly attribute Element? frameElement;
```

```
WindowProxy? open(optional USVString url = "", optional
```

```
DOMString target = "_blank", optional [LegacyNullToEmptyString]
```

```
DOMString features = "");
```

```
getter object (DOMString name);
```

```
// Since this is the global object, the IDL named getter adds a
```

```
NamedPropertiesObject exotic
```

```
// object on the prototype chain. Indeed, this does not make
```

```
the global object an exotic object.
```

```
// Indexed access is taken care of by the WindowProxy exotic
```

```
object.
```

```
// the user agent
```

```
readonly attribute Navigator navigator;
```

```
readonly attribute Navigator clientInformation; // legacy alias
```

```
of .navigator
```

```
readonly attribute boolean originAgentCluster;
```

```
// user prompts
```

```
undefined alert();
```

```
undefined alert(DOMString message);
```

```
boolean confirm(optional DOMString message = "");
```

```
DOMString? prompt(optional DOMString message = "", optional
```

```
DOMString default = "");
```

```
undefined print();
```

```
undefined postMessage(any message, USVString targetOrigin,
```

```
optional sequence<object> transfer = []);
```

```
undefined postMessage(any message, optional
```

```
WindowPostMessageOptions options = {});
```

```
// also has obsolete members
```

```
};
```

```
Window includes GlobalEventHandlers;
```

```
Window includes WindowEventHandlers;
```

```
dictionary WindowPostMessageOptions : StructuredSerializeOptions
```

```
{
```

```
USVString targetOrigin = "/";
```

```
};
```

**window.window**

✓

**window.frames**

✓

**window.self**



Ces attributs renvoient tous *window* .

`window.document`



Renvoie la *fenêtre*Document associée à .

`document.defaultView`



Renvoie l' Windowassocié à *document* , s'il y en a un, ou null sinon.

L' Windowobjet a un **associé Document** , qui est un Documentobjet. Il est défini lors de Windowla création de l'objet et n'est modifié que pendant la navigation à partir du fichier `about:blank` Document .

**Le contexte de navigation**Window de A est le contexte de navigation de son associé . *Il s'agit soit de null, soit d'un* contexte de navigation .Document

**Le navigable**Window de A est le navigable dont le document actif est celui associé à , ou null s'il n'y a pas de navigable de ce type .WindowDocument

Les étapes **window**, **frames**et **self**getter consistent à renvoyer le domaine pertinent de ce domaine .`[[GlobalEnv]].[[GlobalThisValue]]`.

Les étapes du **document** getter consistent à renvoyer ceci associé .Document

*L' Documentobjet associé à un Windowobjet peut changer dans un seul cas : lorsque l' algorithme de navigation crée un nouvel Documentobjet pour la première page chargée dans un contexte de navigation . Dans ce cas précis, l' Windowobjet de la page initiale `about:blank` est réutilisé et obtient un nouvel Document objet.*

Les **defaultView**étapes du getter sont :

1. Si le contexte de navigation de this est null, alors renvoyez null.
2. Renvoie l' objet de ce contexte de navigationWindowProxy .



MDN

Pour des raisons historiques, [Window](#) les objets doivent également avoir une propriété inscriptible, configurable et non énumérable nommée `HTMLDocument` dont la valeur est l' [Document](#) [objet interface](#) .

### 7.2.2.1 Ouverture et fermeture des fenêtres

```
window = window.open([ url [, target [, features ] ] ])
```

✓

Ouvre une fenêtre pour afficher l'*URL* (par défaut "[about:blank](#)") et la renvoie. *target* (par défaut "[\\_blank](#)") donne le nom de la nouvelle fenêtre. Si une fenêtre existe déjà avec ce nom, elle est réutilisée. L'argument *features* peut contenir un [ensemble de jetons séparés par des virgules](#) :

" `noopener` "

" `noreferrer` "

Ceux-ci se comportent de manière équivalente aux types de lien [noopener](#) et [noreferrer](#) sur [les liens hypertexte](#) .

" `popup` "

Encourage les agents utilisateurs à fournir une interface utilisateur de navigateur Web minimale pour la nouvelle fenêtre. (Impacts également le [visible](#) getter sur tous les objets.) [BarProp](#)

```
globalThis.open("https://email.example/message/CA000kFcWW97r8yg=SsWg7GgCmp4suVX9o85y8BvNRqMjuc5PXg", undefined, "noopener,popup");
```

```
window.name [ = value ]
```

✓

Renvoie le nom de la fenêtre.

Peut être défini, pour changer le nom.

```
window.close()
```

✓

Ferme la fenêtre.

```
window.closed
```

✓

Renvoie vrai si la fenêtre a été fermée, faux sinon.

```
window.stop()
```

✓

Annule le chargement du document.

Les **étapes d'ouverture de la fenêtre** , étant donné une chaîne *url* , une chaîne *target* et une chaîne *features* , sont les suivantes :

1. Si le [niveau d'imbrication de terminaison](#) de [la boucle d'événements](#) est différent de zéro, renvoie null.
2. Soit *sourceDocument* l'objet [global d'entrée associé](#)`Document` .
3. Si *la cible* est la chaîne vide, définissez *la cible* sur "`_blank`".
4. Soit *tokenizedFeatures* le résultat des fonctionnalités [de tokenisation](#) .
5. Soit *noopener* et *noreferrer* faux.
6. Si *tokenizedFeatures* [`"noopener"`] [existe](#) , alors :
  1. Définissez *noopener* sur le résultat de [l'analyse de \*tokenizedFeatures\* \[`"noopener"`\] en tant que fonctionnalité booléenne](#) .
  2. [Supprimer](#) *tokenizedFeatures* [`"noopener"`].
7. Si *tokenizedFeatures* [`"noreferrer"`] [existe](#) , alors :
  1. Définissez *noreferrer* sur le résultat de [l'analyse de \*tokenizedFeatures\* \[`"noreferrer"`\] en tant que fonctionnalité booléenne](#) .
  2. [Supprimer](#) *tokenizedFeatures* [`"noreferrer"`].
8. Soit *referrerPolicy* la chaîne vide.
9. Si *noreferrer* est true, définissez *noopener* sur true et *referrerPolicy* sur "`no-referrer`".
10. Soit *targetNavigable* et *windowType* le résultat de l'application [des règles de choix d'une cible](#) navigable donnée , [du nœud navigable](#) de *sourceDocument* et de *noopener* .

S'il existe un agent utilisateur qui prend en charge le contrôle-clic sur un lien pour l'ouvrir dans un nouvel onglet, et que l'utilisateur contrôle-clique sur un élément dont le `onclick` gestionnaire utilise l' `window.open()` API pour ouvrir une page dans un `iframe` élément, l'agent utilisateur peut remplacer la sélection du contexte de navigation cible pour cibler à la place un nouvel onglet.

11. Si *targetNavigable* est null, alors renvoie null.
12. Si *windowType* vaut "`new and unrestricted`" ou "`new with no opener`", alors :
  1. Définissez le contexte [de navigation active](#) de *targetNavigable* [est popup](#) sur le résultat de [la vérification si une fenêtre popup est demandée](#) , étant donné *tokenizedFeatures* .

2. [Configurez les fonctionnalités de contexte de navigation](#) pour le [contexte de navigation actif](#) de `targetNavigable` en fonction de `tokenizedFeatures`. [\[VUE CSSOM\]](#)
3. Soit `urlRecord` l' [enregistrement d'URL](#) `about:blank`.
4. Si `url` n'est pas la chaîne vide, [analysez](#) `url` par rapport à l' [objet de paramètres d'entrée](#) et définissez `urlRecord` sur l' [enregistrement d'URL résultant](#), le cas échéant. Si l' algorithme [d'analyse d'une URL](#) a échoué, lancez un `"_SyntaxError_" DOMException`.
5. Si `urlRecord` [correspond](#) `about:blank`, alors effectuez les [étapes de mise à jour de l'URL et de l'historique](#) en fonction du [document actif](#) et de l'`urlRecord` de `targetNavigable`.

*Ceci est nécessaire dans le cas où l'url est quelque chose comme `about:blank?foo`. Si url est tout simplement `about:blank`, cela ne fera rien.*

6. Sinon, [accédez à](#) `targetNavigable` vers `urlRecord` en utilisant `sourceDocument`, avec [referrerPolicy](#) défini sur `referrerPolicy` et [exceptionsEnabled](#) défini sur `true`.

13. Sinon:

1. Si `url` n'est pas la chaîne vide, alors :
  1. Soit `urlRecord` l' [enregistrement d'URL](#) `about:blank`.
  2. [Analysez](#) l'URL par rapport à l' [objet de paramètres d'entrée](#) et définissez `urlRecord` sur l' [enregistrement d'URL résultant](#), le cas échéant. Si l' algorithme [d'analyse d'une URL](#) a échoué, lancez un `"_SyntaxError_" DOMException`.
  3. [Accédez à](#) `targetNavigable` vers `urlRecord` à l'aide de `sourceDocument`, avec [referrerPolicy](#) défini sur `referrerPolicy` et [exceptionsEnabled](#) défini sur `true`.
2. Si `noopener` est false, alors définissez le contexte de navigation d' [ouverture du contexte de navigation actif](#) de `targetNavigable` sur [le contexte de navigation](#) de `sourceDocument`.

14. Si `noopener` est vrai ou `windowType` est `"new with no opener"`, alors renvoie `null`.

15. Renvoie [l'actif](#) de `targetNavigable`. `WindowProxy`

Les étapes de la méthode consistent à exécuter les [étapes d'ouverture de la fenêtre](#) avec `url`, `target` et `features`. `open(url, target, features)`



Le procédé fournit un mécanisme pour [naviguer dans un contexte de navigation](#) existant ou pour ouvrir et naviguer dans un [contexte de navigation auxiliaire](#) .

---

Pour **tokeniser l' argument *features*** :

1. Soit *tokenizedFeatures* une nouvelle [carte ordonnée](#) .
2. Laissez *la position* pointer au premier point de code des *entités* .
3. [Tant que](#) *la position* n'est pas au-delà de la fin des *fonctionnalités* :
  1. Soit *name* la chaîne vide.
  2. Soit *value* la chaîne vide.
  3. [Collectez une séquence de points de code](#) qui sont [des séparateurs d'entités](#) à partir d' *une position* donnée . Cela saute les séparateurs avant le nom.
  4. [Collectez une séquence de points de code](#) qui ne sont pas [des séparateurs d'entités](#) à partir d' *une position* donnée . Définissez *le nom* sur les caractères collectés, [convertis en minuscules ASCII](#) .
  5. Définissez *name* sur le résultat de la normalisation du *nom* [de la fonctionnalité name](#) .
  6. [Tant que](#) *la position* n'est pas au-delà de la fin des *entités* et que le point de code à *la position* dans *les entités* n'est pas U+003D (=) :
    1. Si le point de code à *la position* dans *les entités* est U+002C (,), ou s'il ne s'agit pas d'un [séparateur d'entités](#) , alors [break](#) .
    2. Avancez *la position* de 1.

*Ceci saute au premier U+003D (=) mais ne saute pas au-delà d'un U+002C (,) ou d'un non-séparateur.*
7. Si le point de code à *la position* dans *les entités* est un [séparateur d'entités](#) :
  1. Tant que *la position* n'est pas au-delà de la fin des *fonctionnalités* et que le point de code à *la position* dans *les fonctionnalités* est un [séparateur de fonctionnalités](#) :
    1. Si le point de code à *la position* dans *les entités* est U+002C (,), alors [break](#) .

2. Avancez la position de 1.

*Cela saute au premier non-séparateur mais ne saute pas au-delà d'un U + 002C (,).*

2. Collectez une séquence de points de code qui ne sont pas des points de code séparateurs d'entités à partir d'une position donnée. Définissez la valeur sur les points de code collectés, convertis en minuscules ASCII.

8. Si *name* n'est pas la chaîne vide, définissez *tokenizedFeatures* [ *name* ] sur *value*.

4. Renvoie *tokenizedFeatures*.

Pour **vérifier si une fonctionnalité de fenêtre est définie**, étant donné *tokenizedFeatures*, *featureName* et *defaultValue*:

1. Si *tokenizedFeatures* [ *featureName* ] existe, renvoie le résultat de l'analyse de *tokenizedFeatures* [ *featureName* ] sous la forme d'une fonctionnalité booléenne.
2. Renvoie la valeur par défaut.

Pour **vérifier si une fenêtre contextuelle est demandée**, étant donné *tokenizedFeatures*:

1. Si *tokenizedFeatures* est vide, alors renvoie false.
2. Si *tokenizedFeatures* ["*popup*"] existe, renvoie le résultat de l'analyse de *tokenizedFeatures* ["*popup*"] sous la forme d'une caractéristique booléenne.
3. Soit *location* le résultat de la vérification si une fonctionnalité de fenêtre est définie, étant donné *tokenizedFeatures*, "*location*" et false.
4. Laissez la barre d'outils être le résultat de la vérification si une fonctionnalité de fenêtre est définie, étant donné *tokenizedFeatures*, "*toolbar*" et false.
5. Si *l'emplacement* et la barre d'outils sont tous les deux faux, alors renvoie vrai.
6. Soit *menubar* le résultat de la vérification si une fonctionnalité de fenêtre est définie, étant donné *tokenizedFeatures*, "*menubar*", et false.
7. Si la barre de menu est fausse, alors retourne vrai.
8. Soit *redimensionnable* le résultat de la vérification si une fonctionnalité de fenêtre est définie, étant donné *tokenizedFeatures*, "*resizable*" et true.
9. Si *redimensionnable* est faux, alors retourne vrai.

10. Soit *les barres de défilement* le résultat de [la vérification si une fonctionnalité de fenêtre est définie](#) , étant donné *tokenizedFeatures* , " *scrollbars*" et false.
11. Si *les barres de défilement* sont fausses, alors renvoie vrai.
12. Soit *status* le résultat de [la vérification si une fonctionnalité de fenêtre est définie](#) , étant donné *tokenizedFeatures* , " *status*" et false.
13. Si *le statut* est faux, alors retourne vrai.
14. Renvoie faux.

Un point de code est un **séparateur de caractéristiques** s'il s'agit d' [un espace blanc ASCII](#) , U+003D (=) ou U+002C (,).

Pour des raisons héritées, il existe des alias de certains noms de fonctionnalités. Pour **normaliser un nom** de fonction name , activez *name* :

```
" screenx"
    Retour " left".

" screeny"
    Retour " top".

" innerwidth"
    Retour " width".

" innerheight"
    Retour " height".
```

#### Rien d'autre

*Nom* de retour .

Pour **analyser une caractéristique booléenne** donnée une *valeur* de chaîne :

1. Si *value* est la chaîne vide, alors retourne true.
  2. Si *la valeur* [est](#) " *yes*", alors renvoie vrai.
  3. Si *la valeur* [est](#) " *true*", alors renvoie vrai.
  4. Soit *parsed* le résultat de [l'analyse de value sous la forme d'un entier](#) .
  5. Si *l'analyse* est une erreur, réglez-la sur 0.
  6. Retourne false si *parsed* vaut 0, et true sinon.
-

Les **name** étapes du getter sont :

1. Si this 's navigable est null, renvoie la chaîne vide.
2. Renvoie le nom cible de cet élément navigable .

Les **name** étapes du setter sont :

1. Si this 's navigable est null, alors return.
2. Définissez le nom de la cible navigable de l' état du document de l'entrée active de l'historique de la session navigable sur la valeur donnée .

*Le nom est réinitialisé lorsque le navigable est navigué vers une autre origine .*

---

Les **close()** étapes de la méthode sont :

1. Soit *thisTraversable* égal à null.
2. Pour chaque traversable de niveau supérieur *traversable* de l'ensemble traversable de niveau supérieur de l'agent utilisateur : si l'objet global pertinent du document actif de *traversable* est égal à this , alors définissez *thisTraversable* sur *traversable* et break .

3. Si *thisTraversable* est null, alors retournez.

*Dans ce cas, la méthode est appelée sur un **Window** qui ne correspond pas à un traversable de niveau supérieur , et donc la fermeture n'est pas autorisée.*

4. Si *thisTraversable* se ferme est vrai, alors retournez.
5. Soit *browserContext* le contexte de navigation actif de *thisTraversable* .
6. Soit *sourceSnapshotParams* le résultat de l'instantané des paramètres de l'instantané source étant donné le document actif de *thisTraversable* .
7. Si toutes les conditions suivantes sont vraies :
  - *thisTraversable* est fermable par un script ;
  - le contexte de navigation de l'objet global en place est familier avec scanningContext ; et
  - le nœud navigable de l' objet global titulaire est autorisé par le sandboxing à naviguer dans *thisTraversable* , étant donné *sourceSnapshotParams*

alors:

- Définissez `thisTraversable` 's [is close](#) sur true.
- [Mettez une tâche en file d'attente](#) sur la [source de la tâche de manipulation DOM](#) pour [fermer](#) `thisTraversable` .

Un [navigable](#) est **fermable par un script** si son [contexte de navigation actif](#) est un [contexte de navigation auxiliaire](#) qui a été créé par un script (par opposition à une action de l'utilisateur), ou s'il s'agit d'un [traversable de niveau supérieur](#) dont [la taille](#) des [entrées de l'historique de session](#) est 1.

Les `closed` étapes du getter consistent à retourner true si [le contexte de navigation](#) de [this](#) est null ou si sa [fermeture](#) est true ; sinon faux.

Les `stop()` étapes de la méthode sont :

1. Si [this](#) 's [navigable](#) est null, alors return.
2. [Arrêtez de](#) charger [c'est](#) [navigable](#) .

#### 7.2.2.2 Accès indexé sur l' [Window](#) objet

`window.length`

✓

Renvoie le nombre de [navigables enfants de l'arborescence de documents](#) .

`window[index]`

Renvoie les navigables enfants [WindowProxy](#) correspondants à l' [arborescence de documents](#) indiquée .

Les `length` étapes du getter consistent à renvoyer [la taille](#) des objets [navigables enfants de l'arborescence de documents](#) `Document` associée .

*L'accès indexé aux [objets navigables enfants de l'arborescence des documents](#) est défini par la méthode interne [\[\[GetOwnProperty\]\]](#) de l' [WindowProxy](#) objet.*

#### 7.2.2.3 Accès nommé sur l' [Window](#) objet

`window[name]`

Renvoie l'élément ou la collection d'éléments indiqué(e).

En règle générale, s'appuyer sur cela conduira à un code fragile. Les identifiants qui finissent par être mappés à cette API peuvent varier au fil du temps, à mesure que de nouvelles fonctionnalités sont ajoutées à la plate-

forme Web, par exemple. Au lieu de cela,  
utilisez `document.getElementById()` OU `document.querySelector()`.

L'ensemble de propriétés de nom de cible navigable enfant de l'arborescence de documents d'une *fenêtre* `Window` d'objet est la valeur de retour de l'exécution de ces étapes :

1. Laissez les enfants être les éléments navigables enfants de l'arborescence de documents associés à la *fenêtre* `.Document`
2. Soit *firstNamedChildren* un ensemble ordonné vide .
3. Pour chaque *navigable* d' *enfants* :
  1. Soit *name* le nom cible de *navigable* .
  2. Si *name* est la chaîne vide, continuez .
  3. Si *noms* contient *nom* , alors continuez .
  4. Ajouter *navigable* à *firstNamedChildren* .
4. Soit *les noms* un ensemble ordonné vide .
5. Pour chaque *navigable* de *firstNamedChildren* :
  1. Soit *name* le nom cible de *navigable* .
  2. Si l'origine du document actif de *navigable* est la même origine que l'origine de l'objet de paramètres pertinent de la *fenêtre* , alors ajoutez le *nom* aux *noms* .
6. Renvoie *les noms* .

Les deux itérations séparées signifient que dans l'exemple suivant, hébergé sur `https://example.org/`, en supposant que `https://elsewhere.example/` la valeur est définie `window.name` sur " *spices*", l'évaluation `window.spices` après que tout a été chargé donnera un résultat indéfini :

```
<iframe src=https://elsewhere.example.com/></iframe>
<iframe name=spices></iframe>
```

L' `Window` objet prend en charge les propriétés nommées . Les noms de propriété pris en charge d'une *fenêtre* `Window` d'objet à tout moment se composent des éléments suivants, dans l'ordre de l'arborescence selon l'élément qui les a contribués, en ignorant les doublons ultérieurs :

- ensemble de propriétés de nom de cible navigable enfant de l'arborescence de documents de la *fenêtre* ;

- la valeur de l' `name` attribut content pour tous les éléments `embed`, `form`, `img` et `object` qui ont un attribut content non vide `name` et sont [dans une arborescence de documents](#) avec des *fenêtres associées* `Document` comme [racine](#) ; et
- la valeur de l' `id` attribut content pour tous [les éléments HTML](#) qui ont un attribut content non vide `id` et qui se trouvent [dans une arborescence de documents](#) avec des *fenêtres associées* `Document` comme [racine](#) .

Pour [déterminer la valeur d'un](#) *nom* de propriété nommé dans une *fenêtre* `Window` d'objet , l'agent utilisateur doit renvoyer la valeur obtenue en procédant comme suit :

1. Soit *objects* la liste des [objets nommés](#) de *window* avec le nom *name* .

*Il y aura au moins un tel objet, par définition.*

2. Si *objects* contient un [navigable](#) , alors :
  1. Soit *conteneur* le premier [conteneur navigable](#) dans les [descendants associés](#) de la *fenêtre* dont [le contenu navigable](#) se trouve dans *les objets* `Document`
  2. Renvoie [le contenu navigable](#) du *conteneur* actif `WindowProxy`
3. Sinon, si *objects* n'a qu'un seul élément, retournez cet élément.
4. Sinon renvoie une `HTMLCollection` *racine* à la *fenêtre associée* `Document` dont le filtre ne correspond qu'aux [objets nommés](#) de *la fenêtre* avec le nom *nom* . (Par définition, ce seront tous des éléments.)

**Les objets nommés** de *la fenêtre* `Window` d'objet avec le nom *name* , aux fins de l'algorithme ci-dessus, se composent des éléments suivants :

- [navigables enfants de l'arborescence des documents](#) des *fenêtres associées* dont `le` `Document` nom `cible` est *name* ;
- `embed`, `form`, `img`, ou `object` des éléments qui ont un `name` attribut de contenu dont la valeur est *name* et qui se trouvent [dans une arborescence de documents](#) avec des *fenêtres associées* `Document` comme [racine](#) ; et
- [Les éléments HTML](#) qui ont un `id` attribut de contenu dont la valeur est *name* et qui se trouvent [dans une arborescence de documents](#) avec des *fenêtres associées* `Document` comme [racine](#) .

#### 7.2.2.4 Accéder aux fenêtres associées

`window.top`



Renvoie le `WindowProxy` pour le [traversable de niveau supérieur](#) .

`window.opener` [ = value ]



Renvoie le `WindowProxy` pour le [contexte de navigation de l'ouvreur](#) .

Renvoie null s'il n'y en a pas ou s'il a été défini sur null.

Peut être défini sur null.

`window.parent`



Renvoie le `WindowProxy` pour le [parent navigable](#) .

`window.frameElement`



Renvoie l'élément [conteneur navigable](#) .

Renvoie null s'il n'y en a pas et dans les situations d'origine croisée.

Les **top** étapes du getter sont :

1. Si [this](#) 's [navigable](#) est null, alors retournez null.
2. Renvoie active [la](#) traversée de [niveau supérieur](#) de [ce](#) navigable `.WindowProxy`

Les **opener** étapes du getter sont :

1. Soit *courant* ce contexte [de navigation](#) .
2. Si *courant* est nul, alors retournez nul.
3. Si [le contexte de navigation](#) de l'ouvreur *actuel* est nul, alors renvoie nul.
4. Renvoie l'objet du [contexte de navigation](#) de l'ouvreur *actuel* `.WindowProxy`

Les **opener** étapes du setter sont :

1. Si la valeur donnée est nulle et que [le contexte de navigation](#) de [this](#) n'est pas nul, alors définissez le contexte de [navigation](#) d'ouverture de [ce contexte de navigation](#) sur null.
2. Si la valeur donnée n'est pas nulle, alors retournez  
? `OrdinaryDefineOwnProperty` ( [this](#) , " opener" , { `[[Value]]` : la valeur donnée, `[[Writable]]` : true, `[[Enumerable]]` : true, `[[Configurable]]` : true } ).

La définition `window.opener` sur null efface la référence [de contexte de navigation de l'ouvreur](#) . En pratique, cela empêche les futurs scripts d'accéder à l'objet de leur contexte [de navigation d'ouverture](#) `Window` .



Par défaut, les scripts peuvent accéder à l'objet de leur [contexte de navigation d'ouverture](#) `window` via le `window.opener` getter. Par exemple, un script peut définir `window.opener.location`, provoquant la [navigation du contexte](#) de navigation de l'ouvreur.

Les **parent** étapes du getter sont :

1. Soit *navigable* c'est *navigable* .
2. Si *navigable* est null, alors retourne null.
3. Si [le parent](#) de *navigable* n'est pas nul, alors définissez *navigable* sur le [parent](#) de *navigable* .
4. Retourne *navigable* 's [active](#) `WindowProxy` .

Les **frameElement** étapes du getter sont :

1. Soit *courant* le [nœud navigable](#) de [ce](#) nœud .
2. Si *courant* est nul, alors retournez nul.
3. Soit *conteneur* le conteneur *actuel* .
4. Si *le conteneur* est nul, alors retournez null.
5. Si [l'origine](#) du [document du nœud](#) du *conteneur* n'est pas [le même domaine d'origine](#) que l' [origine](#) de l'objet de [paramètres actuel](#) , alors renvoyez null.
6. *Conteneur* de retour .

Un exemple de cas où ces propriétés peuvent renvoyer null est le suivant :

```
<!DOCTYPE html>
<iframe></iframe>

<script>
"use strict";
const element = document.querySelector("iframe");
const iframeWindow = element.contentWindow;
element.remove();

console.assert(iframeWindow.top === null);
console.assert(iframeWindow.parent === null);
console.assert(iframeWindow.frameElement === null);
</script>
```

Ici, le [contexte de navigation](#) correspondant à `iframeWindow` a été [annulé](#) lors `element` de sa suppression du document.

### 7.2.2.5 API des éléments d'interface de navigateur historiques

Pour des raisons historiques, l' [Window](#) interface avait certaines propriétés qui représentaient la visibilité de certains éléments de l'interface du navigateur Web.

Pour des raisons de confidentialité et d'interopérabilité, ces propriétés renvoient désormais des valeurs indiquant si la propriété [is popup](#) [Window](#) du contexte [de navigation](#) est true ou false.

Chaque élément d'interface est représenté par un [BarProp](#) objet :

✓ MDN

```
[Exposed=Window]
```

```
interface BarProp {
```

```
    readonly attribute boolean visible;
```

```
};
```

`window.locationbar.visible`

✓

`window.menubar.visible`

✓

`window.personalbar.visible`

✓

`window.scrollbars.visible`

✓

`window.statusbar.visible`

✓

`window.toolbar.visible`

✓

Renvoie vrai si [Window](#) n'est pas une fenêtre contextuelle ; sinon, renvoie faux.

✓ MDN

Les **visible** étapes du getter sont :

1. Soit `browserContext` le [contexte de navigation](#) de [cet](#) objet [global pertinent](#) .

2. Si *scanningContext* est null, alors retourne true.
3. Renvoie la négation du contexte de navigation de [niveau supérieur de scanningContext](#) is [popup](#) .

Les [BarProp](#) objets suivants doivent exister pour chaque [Window](#) objet :

**[BarProp](#) L' objet barre d'adresse**

Historiquement représenté l'élément de l'interface utilisateur qui contient un contrôle qui affiche la barre d'emplacement du navigateur.

**L' [BarProp](#) objet barre de menus**

Historiquement représenté l'élément d'interface utilisateur qui contient une liste de commandes sous forme de menu, ou un concept d'interface similaire.

**[BarProp](#) L' objet bar personnel**

Historiquement représenté l'élément d'interface utilisateur qui contient des liens vers les pages préférées de l'utilisateur, ou un concept d'interface similaire.

**L' [BarProp](#) objet barre de défilement**

Historiquement représenté l'élément d'interface utilisateur qui contient un mécanisme de défilement, ou un concept d'interface similaire.

**[BarProp](#) L' objet barre d'état**

Historiquement représenté un élément d'interface utilisateur trouvé immédiatement en dessous ou après le document, selon le média de l'utilisateur, qui fournit généralement des informations sur l'activité réseau en cours ou des informations sur les éléments que le dispositif de pointage de l'utilisateur indique actuellement.

**L' [BarProp](#) objet barre d'outils**

Historiquement représenté l'élément d'interface utilisateur trouvé immédiatement au-dessus ou avant le document, selon le média de l'utilisateur, qui fournit généralement des contrôles [de parcours de l'historique de session](#) (boutons Précédent et Suivant, boutons de rechargement, etc.).

L' [locationbar](#) attribut doit renvoyer l' [objet barre d'adresse](#)[BarProp](#) .

L' [menubar](#) attribut doit renvoyer l' [objet barre de menus](#)[BarProp](#) .

L' [personalbar](#) attribut doit renvoyer l' [objet barre personnelle](#)[BarProp](#) .

L' [scrollbars](#) attribut doit renvoyer l' [BarProp](#) [objet](#) barre de défilement .

L' [statusbar](#) attribut doit renvoyer l' [objet barre d'état](#)[BarProp](#) .

L' [toolbar](#) attribut doit renvoyer l' [BarProp](#) [objet](#) barre d'outils .

---

Pour des raisons historiques, l' **status** attribut sur l' Window objet doit, lors de l'obtention, renvoyer la dernière chaîne à laquelle il a été défini, et lors de la définition, doit se définir lui-même sur la nouvelle valeur. Lorsque l' Window objet est créé, l'attribut doit être défini sur la chaîne vide. Il ne fait rien d'autre.

#### 7.2.2.6 Paramètres de script pour Window les objets

Pour **configurer un objet de paramètres d'environnement de fenêtre** , étant donné une URL *creationURL* , un *contexte d'exécution* de contexte d'exécution JavaScript , null ou un environnement *réservéEnvironment* , une URL *topLevelCreationURL* et une origine *topLevelOrigin* , exécutez ces étapes :

1. Soit *realm* la valeur du composant Realm du *contexte d'exécution* .
2. Soit *window* l' objet global du *domaine* .
3. Soit *l'objet de paramètres* un nouvel objet de paramètres d'environnement dont les algorithmes sont définis comme suit :

##### **Le contexte d'exécution du domaine**

Renvoie le *contexte d'exécution* .

##### **La carte des modules**

Renvoie la carte des modules de la *fenêtre associée* *Document* .

##### **L' encodage des caractères de l'URL de l'API**

Renvoie l' encodage de caractères courant de la *fenêtre associée* *Document* .

##### **L' URL de base de l'API**

Renvoie l' URL de base actuelle de la *fenêtre associée* *Document* .

##### **L' origine**

Renvoie l' origine de la *fenêtre associée* *Document* .

##### **Le conteneur de politique**

Renvoie le conteneur de politique de la *fenêtre associée* *Document* .

##### **La capacité d'isolement d'origine croisée**

Renvoie true si les deux conditions suivantes sont remplies, et false sinon :

1. le mode d'isolation inter-origine du cluster d' agents du *domaine* est " " , et concrete

2. [La Document](#) *fenêtre* associée est [autorisée à utiliser](#) la [cross-origin-isolated](#) fonctionnalité " " .

### L' [origine du temps](#)

Renvoie [l'heure de début de navigation](#) de [l'information de temps de chargement](#) de [la](#) *fenêtre* associée .[Document](#)

4. Si *l'environnement réservé* n'est pas nul, alors :
  1. Définissez l' [id de l' objet de paramètres](#) sur l' [id](#) de l' *environnement réservé* , [le contexte de navigation cible](#) sur [le contexte de navigation cible](#) de l' *environnement réservé* et [l' agent de service actif](#) sur [l' agent de service actif](#) de l' *environnement réservé* .
  2. Définissez l' [id](#) de l'*environnement réservé* sur la chaîne vide.

*L'identité de l'environnement réservé est considérée comme entièrement transférée à l' [objet de paramètres d'environnement](#) créé . L'environnement réservé ne peut plus être recherché par [l'identifiant](#) de l' [environnement](#) à partir de ce moment.*
5. Sinon, définissez [l' id](#) de l'*objet de paramètres* sur une nouvelle chaîne opaque unique, [le contexte de navigation cible](#) de l'*objet de paramètres* sur null et [le travailleur de service actif](#) de l'*objet de paramètres* sur null.
6. Définissez [l'URL](#) de création de l'*objet de paramètres* sur *creationURL* , [l'URL de création de niveau supérieur](#) de l'*objet de paramètres* sur *topLevelCreationURL* et [l'origine de niveau supérieur](#) de l'*objet de paramètres* sur *topLevelOrigin* .
7. Définissez le *champ* `[[HostDefined]]` du domaine sur l'*objet settings* .

## 7.2.3 L' [WindowProxy](#) objet exotique

A [WindowProxy](#) est un objet exotique qui enveloppe un [Window](#) objet ordinaire, redirigeant la plupart des opérations vers l'objet enveloppé. A chaque [contexte de navigation](#) est associé un [WindowProxy](#) objet. Lors de la [navigation](#) dans le [contexte de navigation](#) , l' objet enveloppé par l' objet associé au [contexte de navigation](#) est modifié.[WindowWindowProxy](#)

L' [WindowProxy](#) objet exotique doit utiliser les méthodes internes ordinaires sauf indication contraire explicite ci-dessous.

Il n'y a pas [WindowProxy](#) [d'objet d'interface](#) .

Chaque [WindowProxy](#) objet a un emplacement interne `[[Window]]`[Window](#) représentant l' objet enveloppé.

*Bien qu'il [WindowProxy](#) soit nommé "proxy", il n'effectue pas de répartition polymorphe sur les méthodes internes de sa cible comme le ferait un vrai proxy, en raison d'un désir de réutiliser les machines entre [WindowProxy](#) et [Location](#) les objets. Tant que l' [Window](#) objet reste un objet ordinaire, cela est inobservable et peut être implémenté dans les deux sens.*

#### 7.2.3.1 **[[GetPrototypeOf]] ( )**

1. Soit *W* la valeur du slot interne [\[\[Window\]\]](#) de **this** .
2. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est vrai, alors retournez ! [OrdinaireGetPrototypeOf](#) ( *W* ).
3. Renvoie nul.

#### 7.2.3.2 **[[SetPrototypeOf]] ( *V* )**

1. Retour ! [SetImmutablePrototype](#) ( **this** , *V* ).

#### 7.2.3.3 **[[EstExtensible]] ( )**

1. Renvoie vrai.

#### 7.2.3.4 **[[Empêcher les extensions]] ( )**

1. Renvoie faux.

#### 7.2.3.5 **[[GetOwnProperty]] ( *P* )**

1. Soit *W* la valeur du slot interne [\[\[Window\]\]](#) de **this** .
2. Si *P* est un [nom de propriété d'index de tableau](#) , alors :
  1. Soit *index* ! [ToUint32](#) ( *P* ).

2. Soit *children* les [éléments navigables enfants de l'arborescence de documents associés](#) à *W.Document*
3. Soit *la valeur* indéfinie.
4. Si *index* est inférieur à [la taille](#) des *enfants* , alors :
  1. [Trier](#) les *enfants* dans l'ordre croissant, *navigableA* étant inférieur à *navigableB* si [le conteneur](#) de *navigableA* a été inséré dans les *W associés* plus tôt que le [conteneur](#) de *navigableB* .*Document*
  2. Définissez *la valeur* sur *children [ index ]*'s [active](#)*WindowProxy* .
5. Si *la valeur* n'est pas définie, alors :
  1. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est vrai, alors retourne *undefined*.
  2. Lancez un ["SecurityError"](#) *DOMException* .
6. Renvoie [PropertyDescriptor](#) { *[[Value]]* : *valeur* , *[[Writable]]* : *false*, *[[Enumerable]]* : *true*, *[[Configurable]]* : *true* }.
3. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est vrai, alors retournez ! [OrdinaryGetOwnProperty](#) ( *W* , *P* ).

*Il s'agit d'une [violation délibérée des invariants de la spécification JavaScript des méthodes internes essentielles pour maintenir la compatibilité avec le contenu Web existant](#). Voir [tc39/ecma262 numéro 672](#) pour plus d'informations. [\[JAVASCRIPT\]](#)*

4. Laissez *la propriété* être [CrossOriginGetOwnPropertyHelper](#) ( *W* , *P* ).
5. Si *propriété* n'est pas indéfinie, alors retourne *propriété* .
6. Si *la propriété* n'est pas définie et que *P* est dans [l'ensemble de propriétés de nom de cible navigable enfant](#) de l'arborescence de documents de *W* , alors :
  1. Soit *value* l' [actif](#)*WindowProxy* de l' [objet nommé](#) de *W* avec le nom *P* .
  2. Renvoie [PropertyDescriptor](#) { *[[Value]]* : *valeur* , *[[Enumerable]]* : *false*, *[[Writable]]* : *false*, *[[Configurable]]* : *true* }.

*La raison pour laquelle les descripteurs de propriété ne sont pas énumérables, malgré cette incompatibilité avec le comportement de même origine, est pour la compatibilité avec le contenu Web existant. Voir [le numéro 3183](#) pour plus de détails.*

7. Retour ? [CrossOriginPropertyFallback](#) ( *P* ).

#### 7.2.3.6 [[DéfinirProperty]] ( *P* , *Desc* )

1. Soit *W* la valeur du slot interne [\[\[Window\]\]](#) de **this** .
2. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est vrai, alors :
  1. Si *P* est un [nom de propriété d'index de tableau](#) , renvoie false.
  2. Retour ? [OrdinaryDefineOwnProperty](#) ( *W* , *P* , *Desc* ).

*Il s'agit d'une violation délibérée des invariants de la spécification JavaScript des méthodes internes essentielles pour maintenir la compatibilité avec le contenu Web existant. Voir [tc39/ecma262 numéro 672](#) pour plus d'informations. [JAVASCRIPT]*

3. Lancez un `"_SecurityError_"_DOMException` .

#### 7.2.3.7 [[Obtenir]] ( *P* , *Récepteur* )

1. Soit *W* la valeur du slot interne [\[\[Window\]\]](#) de **this** .
2. [Vérifier si un accès entre deux contextes de navigation doit être signalé](#) compte tenu du [contexte de navigation](#) de l'[objet global courant](#) , du [contexte de navigation](#) de *W* , *P* , et de l' [objet paramètres courant](#) .
3. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est vrai, alors retournez ? [OrdinaryGet](#) ( **this** , *P* , *Receiver* ).
4. Retour ? [CrossOriginGet](#) ( **this** , *P* , *Receiver* ).

*ceci est passé plutôt que *W* car [OrdinaryGet](#) et [CrossOriginGet](#) invoqueront la méthode interne [\[\[GetOwnProperty\]\]](#) .*

#### 7.2.3.8 [[Régler]] ( *P* , *V* , *Récepteur* )

1. Soit *W* la valeur du slot interne [\[\[Window\]\]](#) de **this** .
2. [Vérifier si un accès entre deux contextes de navigation doit être signalé](#) compte tenu du [contexte de navigation](#) de l'[objet global courant](#) , du [contexte de navigation](#) de *W* , *P* , et de l' [objet paramètres courant](#) .
3. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est vrai, alors :
  1. Si *P* est un [nom de propriété d'index de tableau](#) , alors renvoie false.



2. Retour ? [OrdinarySet](#) ( *W* , *P* , *V* , *Receiver* ).
4. Retour ? [CrossOriginSet](#) ( **this** , *P* , *V* , *Receiver* ).

*ceci est passé plutôt que W car [CrossOriginSet](#) invoquera la méthode interne [\[\[GetOwnProperty\]\]](#) .*

#### 7.2.3.9 [\[\[Supprimer\]\]](#) ( *P* )

1. Soit *W* la valeur du slot interne [\[\[Window\]\]](#) de **this** .
2. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est vrai, alors :
  1. Si *P* est un [nom de propriété d'index de tableau](#) , alors :
    1. Soit *desc* ! **this** .[\[\[GetOwnProperty\]\]](#)( *P* ).
    2. Si *desc* n'est pas défini, alors retourne vrai.
    3. Renvoie faux.
  2. Retour ? [OrdinaireSupprime](#) ( *W* , *P* ).
3. Lancez un ["SecurityError"](#) [DOMException](#) .

#### 7.2.3.10 [\[\[OwnPropertyKeys\]\]](#) ( )

1. Soit *W* la valeur du slot interne [\[\[Window\]\]](#) de **this** .
2. Soit *maxProperties* [la taille](#) des [éléments navigables enfants de](#) [l'](#) arbre de documents [associé](#) à *W*.[Document](#)
3. Soit *keys* la [plage](#) 0 à *maxProperties* , exclusive.
4. Si [IsPlatformObjectSameOrigin](#) ( *W* ) est true, renvoie la concaténation des clés et [OrdinaryOwnPropertyKeys](#) ( *W* ).
5. Renvoie la concaténation des clés et ! [CrossOriginOwnPropertyKeys](#) ( *W* ).

#### 7.2.4 L' [Location](#) interface

Chaque [Window](#) objet est associé à une instance unique d'un [Location](#) objet, allouée lors [Window](#) de la création de l'objet.

**L' [Location](#) objet exotique est défini par un méli-mélo d'IDL, l'invocation de méthodes internes JavaScript après la création et des méthodes internes JavaScript remplacées. Couplé à sa politique de sécurité effrayante, veuillez faire très attention lors de la mise en œuvre de cette excroissance.**

Pour créer un [Location](#) objet, exécutez ces étapes :

1. Soit *location* un nouvel [Location](#) [objet platform](#) .
2. Soit *valueOf* le [domaine pertinent](#) de *location* .[\[\[Intrinsics\]\].\[\[ %Object.prototype.valueOf% \]\]](#).
3. Effectuer ! *location* .[\[\[DefineOwnProperty\]\]](#)(" *valueOf*", { [\[\[Value\]\]](#) : *valueOf* , [\[\[Writable\]\]](#) : false, [\[\[Enumerable\]\]](#) : false, [\[\[Configurable\]\]](#) : false }).
4. Effectuer ! *location* .[\[\[DefineOwnProperty\]\]](#)( [@@toPrimitive](#) , { [\[\[Value\]\]](#) : non défini, [\[\[Writable\]\]](#) : false, [\[\[Enumerable\]\]](#) : false, [\[\[Configurable\]\]](#) : false }).
5. Définissez la valeur de l' emplacement interne [\[\[DefaultProperties\]\]](#) [de](#) l'emplacement sur l'emplacement .[\[\[OwnPropertyKeys\]\]](#)()).
6. *Lieu* de retour .

L'ajout de *valueOf* et [@@toPrimitive](#) propriétés de données propres, ainsi que le fait que tous [Location](#) les attributs IDL de sont marqués , est requis par le code hérité qui a consulté l' interface, ou l'a stringifiée, pour déterminer l' [URL du document](#) , puis l'a utilisé d'une manière sensible à la sécurité. En particulier, les atténuations , [@@toPrimitive](#) et stringifier garantissent que le code tel que ou ne peut pas être mal

```
dirigé. \[LegacyUnforgeable\]LocationvalueOf\[LegacyUnforgeable\]foo[location] =
barlocation + ""
document.location [ = value ]
window.location [ = value ]
```

Renvoie un [Location](#) objet avec l'emplacement de la page actuelle.

Peut être défini, pour naviguer vers une autre page.

Les étapes getter [Document](#) de l'objet **location** consistent à renvoyer l'objet de [cet](#) objet [global pertinent](#) [Location](#) , s'il [est](#) entièrement [actif](#) , et null dans le cas contraire.

Les étapes du getter [Window](#) de l'objet **location** consistent à renvoyer [cet](#) objet [Location](#).

Location Les objets fournissent une représentation de l' URL de leur associé Document, ainsi que des méthodes pour naviguer et recharger le navigable associé .

```
[Exposed=Window]
```

```
interface Location { // but see also additional creation steps
```

```
and overridden internal methods
```

```
[LegacyUnforgeable] stringifier attribute USVString href;
```

```
[LegacyUnforgeable] readonly attribute USVString origin;
```

```
[LegacyUnforgeable] attribute USVString protocol;
```

```
[LegacyUnforgeable] attribute USVString host;
```

```
[LegacyUnforgeable] attribute USVString hostname;
```

```
[LegacyUnforgeable] attribute USVString port;
```

```
[LegacyUnforgeable] attribute USVString pathname;
```

```
[LegacyUnforgeable] attribute USVString search;
```

```
[LegacyUnforgeable] attribute USVString hash;
```

```
[LegacyUnforgeable] undefined assign(USVString url);
```

```
[LegacyUnforgeable] undefined replace(USVString url);
```

```
[LegacyUnforgeable] undefined reload();
```

```
[LegacyUnforgeable, SameObject] readonly attribute
```

```
DOMStringList ancestorOrigins;
```

```
};
```

```
location.toString()
```

```
location.href
```

✓

Renvoie l' Location URL de l'objet.

Peut être défini pour accéder à l'URL donnée.

`location.origin`

✓ 

Renvoie l' [Location](#) origine de l'URL de l'objet.

`location.protocol`

✓ 

Renvoie le [Location](#) schéma de l'URL de l'objet.

Peut être défini pour naviguer vers la même URL avec un schéma modifié.

`location.host`

✓ 

Renvoie l' [Location](#) hôte et le port de l'URL de l'objet (s'il est différent du port par défaut du schéma).

Peut être défini pour naviguer vers la même URL avec un hôte et un port modifiés.

`location.hostname`

✓ 

Renvoie l' [Location](#) hôte de l'URL de l'objet.

Peut être défini pour naviguer vers la même URL avec un hôte modifié.

`location.port`

✓ 

Renvoie le [Location](#) port de l'URL de l'objet.

Peut être défini pour naviguer vers la même URL avec un port modifié.

`location.pathname`

✓ 

Renvoie le [Location](#) chemin de l'URL de l'objet.

Peut être défini pour naviguer vers la même URL avec un chemin modifié.

`location.search`

✓ 

Renvoie la [Location](#) requête de l'URL de l'objet (inclut le " ? " si non vide).

Peut être défini pour naviguer vers la même URL avec une requête modifiée (ignore le " ? ").

`location.hash`

✓ 

Renvoie le [Location](#) fragment d'URL de l'objet (inclut le " #" si non vide).

Peut être défini pour naviguer vers la même URL avec un fragment modifié (ignore le "# ").

```
location.assign(url)
```

✓

Navigue vers l'URL donnée.

```
location.replace(url)
```

✓

Supprime la page actuelle de l'historique de la session et accède à l'URL donnée.

```
location.reload()
```

✓

Recharge la page en cours.

```
location.ancestorOrigins
```

Retourne un `DOMStringList` objet listant les origines des [documents actifs](#) des [navigables ancêtres](#) .

Un `Location` objet a un **pertinent** `Document` associé , qui est [le document actif](#) du [contexte de navigation](#) de son [objet global pertinent](#) , si [le contexte de navigation](#) de [l'objet global pertinent](#) de cet objet est non nul, et nul sinon. `Location`

Un `Location` objet a une **url** associée , qui est [l'URL pertinente](#) de cet `Location` objet , si [la pertinente](#) de cet objet est non nulle, et dans le cas contraire. `DocumentLocationDocumentabout:blank`

Un `Location` objet est associé à une **liste d'origines d'ancêtres** . Lorsqu'un `Location` objet est créé, sa [liste d'origines ancêtre](#) doit être définie sur un `DOMStringList` objet dont la liste associée est la [liste](#) des chaînes que les étapes suivantes produiraient :

1. Soit *la sortie* une nouvelle [liste](#) de chaînes.
2. Soit *courant* l' `Location` élément [pertinent](#) `Document` de l'objet .
3. Alors que le [document conteneur](#) *actuel* n'est pas nul :
  1. Définit *current* sur le [document conteneur](#) *actuel* .
  2. [Ajouter](#) la [sérialisation](#) de [l'origine](#) du *courant* à *la sortie* .
4. *Sortie* de retour .

Pour **Location-object** naviguer un emplacement Location d'objet vers une URL url , éventuellement donné un comportement de gestion de l'historique *historyHandling* (par défaut " "): push

1. Soit *navigable* le navigable de l' objet global pertinent de l'emplacement .
2. Soit *sourceDocument* l' objet global titulaire associé *Document* .
3. Si l' emplacement pertinent *Document* n'est pas encore complètement chargé et que l' objet global titulaire n'a pas d' activation transitoire , définissez *historyHandling* sur " replace " .
4. Naviguez *navigable* vers l'*url* en utilisant *sourceDocument* , avec exceptionsEnabled défini sur true et historyHandling défini sur *historyHandling* .

Les **href** étapes du getter sont :

1. Si cela est pertinent *Document* n'est pas nul et que son origine n'est pas le même domaine d' origine que l' origine de l' objet de paramètres d' entrée , lancez alors un " " *.SecurityError DOMException*
2. Renvoie cette URL , sérialisée . \_

Les **href** étapes du setter sont :

1. Si cela est pertinent *Document* est null, alors retournez.
2. Analyse la valeur donnée par rapport à l' objet de paramètres d'entrée . Si cela échoue, lancez un " *SyntaxError* " *DOMException* .
3. **Location-object** navigue jusqu'à l' enregistrement d'URL résultant .

*Le href passeur n'a intentionnellement aucun contrôle de sécurité.*

Les **origin** étapes du getter sont :

1. Si cela est pertinent *Document* n'est pas nul et que son origine n'est pas le même domaine d' origine que l' origine de l' objet de paramètres d' entrée , lancez alors un " " *.SecurityError DOMException*
2. Renvoie la sérialisation de l' origine de cette URL .

Les **protocol** étapes du getter sont :

1. Si cela est pertinent *Document* n'est pas nul et que son origine n'est pas le même domaine d' origine que l' origine de l' objet de paramètres d' entrée , lancez alors un " " *.SecurityError DOMException*
2. Renvoie le schéma de cette URL , suivi de " " . :

Les protocol étapes du setter sont :

1. Si cela est pertinent`Document` est null, alors retournez.
2. Si l'origine de cet objet pertinent`Document` n'est pas le même domaine d'origine que l'origine de l'objet des paramètres d'entrée, lancez un `"".SecurityError DOMException`
3. Soit `copyURL` une copie de cette URL .
4. Soit `possibleFailure` le résultat de l'analyse de l'URL de base de la valeur donnée, suivi de " :", avec `copyURL` comme url et l'état de début du schéma comme état override .

*Étant donné que l'analyseur d'URL ignore plusieurs deux-points consécutifs, fournir une valeur de "https:" (ou même "https:::") équivaut à fournir une valeur de "https".*

5. Si `possibleFailure` est un échec, lancez un `"SyntaxError" DOMException` .
6. Si le schéma de `copyURL` n'est pas un schéma HTTP(S), terminez ces étapes.
7. `Location-object naviguez` jusqu'à `copyURL` .

Les host étapes du getter sont :

1. Si cela est pertinent`Document` n'est pas nul et que son origine n'est pas le même domaine d'origine que l'origine de l'objet de paramètres d'entrée, lancez alors un `"".SecurityError DOMException`
2. Soit `url` l'url de celle-ci .
3. Si l'hôte de l'`url` est nul, renvoie la chaîne vide.
4. Si le port de l'`url` est nul, renvoie l'hôte de l'`url`, sérialisé .
5. Renvoie l'hôte de l'`url`, sérialisé, suivi de " " et le port de l'`url`, sérialisé .:

Les host étapes du setter sont :

1. Si cela est pertinent`Document` est null, alors retournez.
2. Si l'origine de cet objet pertinent`Document` n'est pas le même domaine d'origine que l'origine de l'objet des paramètres d'entrée, lancez un `"".SecurityError DOMException`
3. Soit `copyURL` une copie de cette URL .
4. Si `copyURL` a un chemin opaque, alors retournez.

5. [L'URL de base analyse](#) la valeur donnée, avec *copyURL* comme [url](#) et [host state](#) comme [state override](#) .
6. *Location-object naviguez* jusqu'à *copyURL* .

Les *hostname* étapes du getter sont :

1. Si [cela](#) est [pertinent](#)*Document* n'est pas nul et que son [origine](#) n'est pas [le même domaine d'origine](#) que l' [origine](#) de l' [objet de paramètres d'entrée](#) , lancez alors un `" "` *.SecurityError DOMException*
2. Si l' [hôte](#) de [cette URL](#) est nul, renvoie la chaîne vide.
3. Renvoie l' [hôte](#) de [cette URL](#) , [sérialisé](#) .

Les *hostname* étapes du setter sont :

1. Si [cela](#) est [pertinent](#)*Document* est null, alors retournez.
2. Si l' [origine](#) de [cet](#) objet [pertinent](#)*Document* n'est pas [le même domaine d'origine](#) que l' [origine](#) de l' [objet des paramètres d'entrée](#) , lancez un `" "` *.SecurityError DOMException*
3. Soit *copyURL* une copie [de cette](#) URL .
4. Si *copyURL* a un [chemin opaque](#) , alors retournez.
5. [L'URL de base analyse](#) la valeur donnée, avec *copyURL* comme [url](#) et [l'état du nom d'hôte](#) comme [état override](#) .
6. *Location-object naviguez* jusqu'à *copyURL* .

Les *port* étapes du getter sont :

1. Si [cela](#) est [pertinent](#)*Document* n'est pas nul et que son [origine](#) n'est pas [le même domaine d'origine](#) que l' [origine](#) de l' [objet de paramètres d'entrée](#) , lancez alors un `" "` *.SecurityError DOMException*
2. Si le [port](#) de [cette URL](#) est nul, renvoie la chaîne vide.
3. Renvoie [le](#) port de [cette URL](#) , [sérialisé](#) .

Les *port* étapes du setter sont :

1. Si [cela](#) est [pertinent](#)*Document* est null, alors retournez.
2. Si l' [origine](#) de [cet](#) objet [pertinent](#)*Document* n'est pas [le même domaine d'origine](#) que l' [origine](#) de l' [objet des paramètres d'entrée](#) , lancez un `" "` *.SecurityError DOMException*
3. Soit *copyURL* une copie [de cette](#) URL .



4. Si *copyURL* ne peut pas avoir de nom d'utilisateur/mot de passe/port , alors retournez.
5. Si la valeur donnée est la chaîne vide, définissez le port de *copyURL* sur null.
6. Sinon, l'URL de base analyse la valeur donnée, avec *copyURL* comme url et port state comme state override .
7. *Location-object naviguez* jusqu'à *copyURL* .

Les *pathname* étapes du getter sont :

1. Si cela est pertinent*Document* n'est pas nul et que son origine n'est pas le même domaine d'origine que l' origine de l' objet de paramètres d'entrée , lancez alors un " " *.SecurityError DOMException*
2. Renvoie le résultat du chemin d'URL sérialisant l'url de cet *Location* objet .

Les *pathname* étapes du setter sont :

1. Si cela est pertinent*Document* est null, alors retournez.
2. Si l' origine de cet objet pertinent*Document* n'est pas le même domaine d'origine que l' origine de l' objet des paramètres d'entrée , lancez un " " *.SecurityError DOMException*
3. Soit *copyURL* une copie de cette URL .
4. Si *copyURL* a un chemin opaque , alors retournez.
5. Définissez le chemin de *copyURL* sur la liste vide.
6. L'URL de base analyse la valeur donnée, avec *copyURL* comme url et path start state comme state override .
7. *Location-object naviguez* jusqu'à *copyURL* .

Les *search* étapes du getter sont :

1. Si cela est pertinent*Document* n'est pas nul et que son origine n'est pas le même domaine d'origine que l' origine de l' objet de paramètres d'entrée , lancez alors un " " *.SecurityError DOMException*
2. Si la requête de cette URL est nulle ou la chaîne vide, renvoie la chaîne vide.
3. Renvoie " ? " , suivi de la requête de cette URL .

Les *search* étapes du setter sont :

1. Si cela est pertinent*Document* est null, alors retournez.

2. Si l'origine de cet objet pertinent`Document` n'est pas le même domaine d'origine que l' 'origine de l' objet des paramètres d'entrée , lancez un `"".SecurityError DOMException`
3. Soit `copyURL` une copie de cette URL .
4. Si la valeur donnée est la chaîne vide, définissez la requête de `copyURL` sur null.
5. Sinon, exécutez ces sous-étapes :
  1. Soit *l'entrée* la valeur donnée avec un seul " ? " supprimé, le cas échéant.
  2. Définissez la requête de `copyURL` sur la chaîne vide.
  3. *Entrée d'analyse d'URL de base* , avec null , le codage de caractères du document `pertinent` `Document` `copyURL` comme url et l' état de la requête comme remplacement d'état .
6. `Location`-object naviguez jusqu'à `copyURL` .

Les hashétapes du getter sont :

1. Si cela est pertinent`Document` n'est pas nul et que son origine n'est pas le même domaine d'origine que l' 'origine de l' objet de paramètres d'entrée , lancez alors un `"".SecurityError DOMException`
2. Si le fragment de cette url est nul ou une chaîne vide, renvoie la chaîne vide.
3. Renvoie " # " , suivi du fragment de cette URL .

Les hashétapes du setter sont :

1. Si cela est pertinent`Document` est null, alors retournez.
2. Si l'origine de cet objet pertinent`Document` n'est pas le même domaine d'origine que l' 'origine de l' objet des paramètres d'entrée , lancez un `"".SecurityError DOMException`
3. Soit `copyURL` une copie de cette URL .
4. Soit *l'entrée* la valeur donnée avec un seul " # " supprimé, le cas échéant.
5. Définissez le fragment de `copyURL` sur la chaîne vide.
6. *Entrée d'analyse d'URL de base* , avec `copyURL` comme URL et état du fragment comme remplacement d'état .
7. Si le fragment de `copyURL` est le fragment de cette URL , alors retournez.

*Ce renflouement est nécessaire pour la compatibilité avec le contenu déployé, qui est défini de manière redondante `location.hash` sur scroll . Elle ne s'applique pas aux autres mécanismes de navigation par fragment, tels que le `location.href` setter ou `location.assign()` .*

8. `Location`-object naviguez jusqu'à `copyURL` .

*Contrairement à l'API équivalente pour les éléments `a` et `area`, le `hash` setter ne casse pas spécialement la chaîne vide, pour rester compatible avec les scripts déployés.*

---

Les étapes de la méthode sont : `assign(url)`

1. Si cela est pertinent `Document` est null, alors retournez.
2. Si l' origine de cet objet pertinent `Document` n'est pas le même domaine d'origine que l' origine de l' objet des paramètres d'entrée , lancez un `"` `.SecurityError` `DOMException` .
3. URL d'analyse relative à l' objet de paramètres d'entrée . Si cela échoue, lancez un `"` `SyntaxError` `DOMException` .
4. `Location`-object navigue jusqu'à l' enregistrement d'URL résultant .

Les étapes de la méthode sont : `replace(url)`

1. Si cela est pertinent `Document` est null, alors retournez.
2. URL d'analyse relative à l' objet de paramètres d'entrée . Si cela échoue, lancez un `"` `SyntaxError` `DOMException` .
3. `Location`-object naviguer jusqu'à l' enregistrement d'URL résultant donné `"` `replace` `"` .

*La `replace()` méthode n'a intentionnellement aucun contrôle de sécurité.*

Les `reload()` étapes de la méthode sont :

1. Que le *document* soit pertinent . `__Document`
2. Si *le document* est nul, alors retournez.
3. Si l' origine du *document* n'est pas le même domaine d'origine que l' origine de l' objet des paramètres d'entrée , lancez un `"` `.SecurityError` `DOMException` .

4. [Recharger](#) le nœud du *document* [navigable](#) .
- 

Les `ancestorOrigins` étapes du getter sont :

1. Si [cela](#) est [pertinent](#)`Document` est null, alors retourne une [liste](#) vide .
2. Si l' [origine](#) de [cet](#) objet [pertinent](#)`Document` n'est pas [le même domaine d'origine](#) que l' [origine](#) de l' [objet des paramètres d'entrée](#) , lancez un `" "` `.SecurityError DOMException`
3. Sinon, renvoie la liste des origines des [ancêtres de this](#) .

**Les détails du `ancestorOrigins` fonctionnement de l'attribut sont encore controversés et pourraient changer. Voir [le numéro 1918](#) pour plus d'informations.**

---

Comme expliqué précédemment, l' `Location` objet exotique nécessite une logique supplémentaire au-delà de l'IDL pour des raisons de sécurité. L' `Location` objet doit utiliser les méthodes internes ordinaires sauf indication contraire explicite ci-dessous.

De plus, chaque `Location` objet a un emplacement interne **[[DefaultProperties]]** représentant ses propres propriétés au moment de sa création.

#### 7.2.4.1 **[[GetPrototypeOf]] ( )**

1. Si [IsPlatformObjectSameOrigin](#) ( **this** ) est vrai, alors return [! OrdinaryGetPrototypeOf](#) ( **this** ).
2. Renvoie nul.

#### 7.2.4.2 **[[SetPrototypeOf]] ( V )**

1. Retour [! SetImmutablePrototype](#) ( **this** , V ).

#### 7.2.4.3 **[[EstExtensible]]** ( )

1. Renvoie vrai.

#### 7.2.4.4 **[[Empêcher les extensions]]** ( )

1. Renvoie faux.

#### 7.2.4.5 **[[GetOwnProperty]]** ( *P* )

1. Si [IsPlatformObjectSameOrigin](#) ( **this** ) est vrai, alors :
  1. Soit *desc* être [OrdinaryGetOwnProperty](#) ( **this** , *P* ).
  2. Si la valeur de l' emplacement interne [\[\[DefaultProperties\]\]](#) de **this** contient *P* , alors définissez *desc* .[[Configurable]] sur true.
  3. Retour *desc* .
2. Laissez *la propriété* être [CrossOriginGetOwnPropertyHelper](#) ( **this** , *P* ).
3. Si *propriété* n'est pas indéfinie, alors retourne *propriété* .
4. Retour ? [CrossOriginPropertyFallback](#) ( *P* ).

#### 7.2.4.6 **[[DéfinirProperty]]** ( *P* , *Desc* )

1. Si [IsPlatformObjectSameOrigin](#) ( **this** ) est vrai, alors :
  1. Si la valeur du slot interne [\[\[DefaultProperties\]\]](#) de **this** contient *P* , alors retourne false.
  2. Retour ? [OrdinaryDefineOwnProperty](#) ( **this** , *P* , *Desc* ).
2. Lancez un `"_SecurityError_"` [DOMException](#) .

#### 7.2.4.7 **[[Obtenir]]** ( *P* , *Récepteur* )

1. Si [IsPlatformObjectSameOrigin](#) ( **this** ) est vrai, alors retournez ? [OrdinaryGet](#) ( **this** , *P* , *Receiver* ).
2. Retour ? [CrossOriginGet](#) ( **this** , *P* , *Receiver* ).

#### 7.2.4.8 [[Régler]] ( *P* , *V* , *Récepteur* )

1. Si [IsPlatformObjectSameOrigin](#) ( **this** ) est vrai, alors retournez ? [OrdinarySet](#) ( **this** , *P* , *V* , *Receiver* ).
2. Retour ? [CrossOriginSet](#) ( **this** , *P* , *V* , *Receiver* ).

#### 7.2.4.9 [[Supprimer]] ( *P* )

1. Si [IsPlatformObjectSameOrigin](#) ( **this** ) est vrai, alors retournez ? [OrdinaryDelete](#) ( **this** , *P* ).
2. Lancez un `"_SecurityError_"` [DOMException](#) .

#### 7.2.4.10 [[OwnPropertyKeys]] ( )

1. Si [IsPlatformObjectSameOrigin](#) ( **this** ) est vrai, alors retournez [OrdinaryOwnPropertyKeys](#) ( **this** ).
2. Renvoie [CrossOriginOwnPropertyKeys](#) ( **this** ).

### 7.2.5 L' [History](#) interface



```
enum ScrollRestoration { "auto", "manual" };
```

```
[Exposed=Window]
```

```
interface History {
```

```
readonly attribute unsigned long length;
```

```
attribute ScrollRestoration scrollRestoration;
```

```
readonly attribute any state;
```

```
undefined go(optional long delta = 0);
```

```
undefined back();
```

```
undefined forward();
```

```
undefined pushState(any data, DOMString unused, optional
```

```
USVString? url = null);
```

```
undefined replaceState(any data, DOMString unused, optional
```

```
USVString? url = null);
```

```
};
```

#### history.length

✓ 

Renvoie le nombre d' [entrées d'historique de session](#) globales pour le [navigable traversable](#) actuel .

#### history.scrollRestoration

✓ 

Renvoie le [mode de restauration du défilement](#) de [l'entrée active de l'historique de la session](#) .

#### history.scrollRestoration = value

Définissez le [mode de restauration du défilement](#) de [l'entrée d'historique de session active](#) sur *la valeur* .

#### history.state

✓ 

Renvoie l' [état sérialisé](#) de [l'entrée d'historique de session active](#) , désérialisé en une valeur JavaScript.

#### history.go()

Recharge la page en cours.

#### history.go(delta)

✓ 

Remonte ou avance le nombre d'étapes spécifié dans la liste globale [des entrées de l'historique de session](#) pour le [navigable traversable](#) actuel .

Un delta nul rechargera la page en cours.

Si le delta est hors plage, ne fait rien.

`history.back()`



Remonte d'une étape dans la liste globale [des entrées de l'historique de session](#) pour le [navigable traversable](#) actuel .

S'il n'y a pas de page précédente, ne fait rien.

`history.forward()`



Avance d'une étape dans la liste globale [des entrées de l'historique de session](#) pour le [navigable traversable](#) actuel .

S'il n'y a pas de page suivante, ne fait rien.

`history.pushState(data, "")`



Ajoute une nouvelle entrée dans l'historique de session avec son [état sérialisé](#) défini sur une sérialisation de *données* . L' [URL](#) de l' [entrée d'historique active](#) sera copiée et utilisée pour l'URL de la nouvelle entrée.

(Le deuxième paramètre existe pour des raisons historiques et ne peut pas être omis ; le passage de la chaîne vide est traditionnel.)

`history.pushState(data, "", url)`

Ajoute une nouvelle entrée dans l'historique de session avec son [état sérialisé](#) défini sur une sérialisation de *données* et avec son [URL](#) définie sur *url* .

Si le courant [Document](#) [ne peut pas avoir son URL réécrite](#) en *url* , un `"_SecurityError"` `DOMException` sera lancé.

(Le deuxième paramètre existe pour des raisons historiques et ne peut pas être omis ; le passage de la chaîne vide est traditionnel.)

`history.replaceState(data, "")`



Met à jour l' [état sérialisé](#) de l' [entrée d'historique de session active](#) en un clone structuré de *données* .

(Le deuxième paramètre existe pour des raisons historiques et ne peut pas être omis ; le passage de la chaîne vide est traditionnel.)

`history.replaceState(data, "", url)`

Met à jour l' [état sérialisé](#) de l' [entrée d'historique de la session active](#) en un clone structuré de *données* et son [URL](#) en *url* .



Si le courant `Document` ne peut pas avoir son URL réécrite en `url` ,  
un `"_".DocumentSecurityError` `DOMException` sera lancé.

(Le deuxième paramètre existe pour des raisons historiques et ne peut pas être omis ; le passage de la chaîne vide est traditionnel.)

A `Document` a un **objet historique** , un `History` objet.

Les **history** étapes du getter consistent à renvoyer l'objet d' historique de `cet associé` `Document` .

---

Chaque `History` objet a **state** , initialement null.

Chaque `History` objet a une **longueur** , un entier non négatif, initialement 0.

Chaque `History` objet a un **index** , un entier non négatif, initialement 0.

*Bien que l' `index` ne soit pas directement exposé, il peut être déduit des modifications de la `longueur` lors des navigations synchrones. En fait, c'est à ça qu'il sert.*

Les **length** étapes du getter sont :

1. Si `cet` objet `global pertinent` associé n'est pas `entièrement actif` , lancez un `"_".DocumentSecurityError` `DOMException`
2. Renvoie `cette` longueur . `_`

Les **scrollRestoration** étapes du getter sont :

1. Si `cet` objet `global pertinent` associé n'est pas `entièrement actif` , lancez un `"_".DocumentSecurityError` `DOMException`
2. Renvoie `le mode de restauration de défilement` de l'entrée d' `historique de session active` de ce nœud `navigable` .

Les **scrollRestoration** étapes du setter sont :

1. Si `cet` objet `global pertinent` associé n'est pas `entièrement actif` , lancez un `"_".DocumentSecurityError` `DOMException`
2. Définissez `le mode de restauration du défilement` de `l'entrée d'historique de session active` de ce `nœud` navigable sur la valeur donnée.

Les **state** étapes du getter sont :

1. Si cet objet global pertinent associé n'est pas entièrement actif , lancez un `"_".DocumentSecurityError DOMException`
2. Renvoie cet état . \_

Les étapes de la méthode consistent à delta traverser ce *delta* donné .`go(delta)`

Les `back()` étapes de la méthode consistent à parcourir en delta ce -1 donné.

Les `forward()` étapes de la méthode consistent à parcourir delta ce +1 donné.

Pour **effectuer une traversée delta** d'un *historique* `History` d'objet étant donné un entier *delta* :

1. Soit *document* l' objet global pertinent de l' *historique* associé .`Document`
2. Si le *document* n'est pas entièrement actif , lancez un `"_SecurityError" DOMException` .
3. Si *delta* vaut 0, alors rechargez le nœud navigable du *document* .
4. Parcourir l'historique par un delta donné par le nœud navigable du *document* navigable , le *delta* et le *document* .

Les étapes de la méthode consistent à exécuter les étapes d'état push/replace de l'historique partagé en fonction de this , *data* , *url* et `"_".pushState(data, unused, url)push`

Les étapes de la méthode consistent à exécuter les étapes d'état push/replace de l'historique partagé en fonction de this , *data* , *url* et `"_".replaceState(data, unused, url)replace`

Les **étapes d'état push/replace de l'historique partagé** , étant donné un `History` *history* , une valeur *data* , une chaîne de valeur scalaire -or-null *url* et un comportement de gestion de l'historique *historyHandling* , sont :

1. Soit le *document* associé à l' *historique* `Document` .
2. Si le *document* n'est pas entièrement actif , lancez un `"_SecurityError" DOMException` .
3. En option, retour. (Par exemple, l'agent utilisateur peut interdire les appels à ces méthodes qui sont invoquées sur une minuterie, ou à partir d'écouteurs d'événements qui ne sont pas déclenchés en réponse à une action claire de l'utilisateur, ou qui sont invoqués en succession rapide.)
4. Laissez *serializedData* être StructuredSerializeForStorage ( *data* ). Renvoyez toutes les exceptions.
5. Soit *newURL* l' URL du *document* .

6. Si l'URL n'est pas nulle, alors :
  1. URL d'analyse , relative à l' objet de paramètres pertinent de l'historique .
  2. Si cela échoue, lancez un `"_SecurityError_" DOMException` .
  3. Définissez `newURL` sur l' enregistrement d'URL résultant .
  4. Si le document ne peut pas avoir son URL réécrite en `newURL` , lancez un `"_SecurityError_" DOMException` .
7. Exécutez les étapes de mise à jour de l'URL et de l'historique en fonction de `document` et de `newURL` , avec `serializedData` défini sur `serializedData` et `historyHandling` défini sur `historyHandling` .

Les agents utilisateurs peuvent limiter le nombre d'objets d'état ajoutés à l'historique de session par page. Si une page atteint la limite définie par l'implémentation , les agents utilisateurs doivent supprimer l'entrée immédiatement après la première entrée pour cet `Document` objet dans l'historique de session après avoir ajouté la nouvelle entrée. (Ainsi, l'historique d'état agit comme un tampon FIFO pour l'éviction, mais comme un tampon LIFO pour la navigation.)

Un `Document` `document` **peut voir son URL réécrite** en URL cible si l'algorithme suivant renvoie vrai :

1. Soit `documentURL` l' URL du `document` .
2. Si `targetURL` et `documentURL` diffèrent dans leurs composants de schéma , nom d'utilisateur , mot de passe , hôte ou port , alors renvoie false.
3. Si le schéma de `targetURL` est un schéma HTTP(S) , alors retourne true. (Les différences de path , query et fragment sont autorisées pour les URL et `http:https:`.)
4. Si le schéma de `targetURL` est " " , et que `targetURL` et `documentURL` diffèrent dans leur composant de chemin , alors renvoie false. (Les différences de requête et de fragment sont autorisées pour les URL.)
5. Si `targetURL` et `documentURL` diffèrent dans leur composant de chemin ou leurs composants de requête , alors renvoie false. (Seules les différences de fragment sont autorisées pour les autres types d'URL.)
6. Renvoie vrai.

<u>URL</u> du document	Cible URL	peut avoir son URL réécrite
https://example.com/home	https://example.com/home#about	✓
https://example.com/home	https://example.com/home?page=shop	✓
https://example.com/home	https://example.com/shop	✓
https://example.com/home	https://user:pass@example.com/home	✗
https://example.com/home	http://example.com/home	✗
file:///path/to/x	file:///path/to/x#hash	✓
file:///path/to/x	file:///path/to/x?search	✓
file:///path/to/x	file:///path/to/y	✗
about:blank	about:blank#hash	✓
about:blank	about:blank?search	✗
about:blank	about:srcdoc	✗
data:text/html,foo	data:text/html,foo#hash	✓
data:text/html,foo	data:text/html,foo?search	✗
data:text/html,foo	data:text/html,bar	✗
data:text/html,foo	data:bar	✗
blob:https://example.com/77becafe-657b-4fdc-8bd3-e83aaa5e8f43	blob:https://example.com/77becafe-657b-4fdc-8bd3-e83aaa5e8f43#hash	✓
blob:https://example.com/77becafe-657b-4fdc-8bd3-e83aaa5e8f43	blob:https://example.com/77becafe-657b-4fdc-8bd3-e83aaa5e8f43?search	✗
blob:https://example.com/77becafe-657b-4fdc-8bd3-e83aaa5e8f43	blob:https://example.com/anything	✗
blob:https://example.com/77becafe-657b-4fdc-8bd3-e83aaa5e8f43	blob:path	✗

Notez que seule l' URL du Document fichier compte, et non son origine . Ils peuvent ne pas correspondre dans des cas tels que about:blank Documents avec des origines héritées, dans iframe des s en bac à sable ou lorsque le document.domain setter a été utilisé.

Considérons un jeu où l'utilisateur peut naviguer le long d'une ligne, de sorte que l'utilisateur soit toujours à une certaine coordonnée, et tel que l'utilisateur puisse mettre en signet la page correspondant à une coordonnée particulière, pour y revenir plus tard.

Une page statique implémentant la position  $x=5$  dans un tel jeu pourrait ressembler à ceci :

```
<!DOCTYPE HTML>
<!-- this is https://example.com/line?x=5 -->
<html lang="en">
<title>Line Game - 5</title>
<p>You are at coordinate 5 on the line.</p>
<p>
  <a href="?x=6">Advance to 6</a> or
  <a href="?x=4">retreat to 4</a>?
</p>
```

Le problème avec un tel système est qu'à chaque fois que l'utilisateur clique, toute la page doit être rechargée. Voici à la place une autre façon de le faire, en utilisant un script :

```
<!DOCTYPE HTML>
<!-- this starts off as https://example.com/line?x=5 -->
<html lang="en">
<title>Line Game - 5</title>
<p>You are at coordinate <span id="coord">5</span> on the line.</p>
<p>
  <a href="?x=6" onclick="go(1); return false;">Advance to 6</a> or
  <a href="?x=4" onclick="go(-1); return false;">retreat to 4</a>?
</p>
<script>
  var currentPage = 5; // prefilled by server
  function go(d) {
    setupPage(currentPage + d);
    history.pushState(currentPage, "", '?x=' + currentPage);
  }
  onpopstate = function(event) {
    setupPage(event.state);
  }
  function setupPage(page) {
    currentPage = page;
    document.title = 'Line Game - ' + currentPage;
    document.getElementById('coord').textContent = currentPage;
    document.links[0].href = '?x=' + (currentPage+1);
    document.links[0].textContent = 'Advance to ' + (currentPage+1);
    document.links[1].href = '?x=' + (currentPage-1);
```

```
document.links[1].textContent = 'retreat to ' + (currentPage-1);
}
</script>
```

Dans les systèmes sans script, cela fonctionne toujours comme l'exemple précédent. Cependant, les utilisateurs qui *prennent* en charge les scripts peuvent désormais naviguer beaucoup plus rapidement, car il n'y a pas d'accès au réseau pour la même expérience. De plus, contrairement à l'expérience que l'utilisateur aurait avec une approche basée sur des scripts naïfs, la mise en signet et la navigation dans l'historique de la session fonctionnent toujours.

Dans l'exemple ci-dessus, l'argument *data* de la [pushState\(\)](#) méthode est la même information que celle qui serait envoyée au serveur, mais sous une forme plus pratique, de sorte que le script n'ait pas à analyser l'URL à chaque fois que l'utilisateur navigue.

La plupart des applications souhaitent utiliser la même valeur [de mode de restauration de défilement](#) pour toutes leurs entrées d'historique. Pour ce faire, ils peuvent définir l' [scrollRestoration](#) attribut dès que possible (par exemple, dans le premier [script](#) élément de l'élément du document [head](#)) pour s'assurer que toute entrée ajoutée à la session d'historique obtient le mode de restauration de défilement souhaité.

```
<head>
  <script>
    if ('scrollRestoration' in history)
      history.scrollRestoration = 'manual';
  </script>
</head>
```

## 7.2.6 Interfaces d'événements

### 7.2.6.1 L' [PopStateEvent](#) interface



```
[Exposed=Window]
```

```
interface PopStateEvent : Event {
```

```
  constructor(DOMString type, optional PopStateEventInit
```

```
  eventInitDict = {});
```

```
readonly attribute any state;
```

```
};
```

```
dictionary PopStateEventInit : EventInit {
```

```
any state = null;
```

```
};
```

#### **event.state**

Renvoie une copie des informations fournies  
à [pushState\(\)](#) ou [replaceState\(\)](#).

L' **state** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente les informations de contexte pour l'événement, ou null, si l'état représenté est l'état initial du [Document](#).

### 7.2.6.2 L' [HashChangeEvent](#) interface



MDN

```
[Exposed=Window]
```

```
interface HashChangeEvent : Event {
```

```
constructor(DOMString type, optional HashChangeEventInit
```

```
eventInitDict = {});
```

```
readonly attribute USVString oldURL;
```

```
readonly attribute USVString newURL;
```

```
};
```

```
dictionary HashChangeEventInit : EventInit {
```

```
USVString oldURL = "";
```

```
USVString newURL = "";
```

```
};
```

**event.oldURL**

✓

Renvoie l' [URL](#) de l' [entrée d'historique de session](#) qui était précédemment en cours.

**event.newURL**

✓

Renvoie l' [URL](#) de l' [entrée d'historique de session](#) qui est actuellement en cours.

L' **oldURL** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente des informations de contexte pour l'événement, en particulier l'URL de l' [entrée d'historique de session](#) à partir de laquelle il a été parcouru.

L' **newURL** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente les informations contextuelles de l'événement, en particulier l'URL de l' [entrée de l'historique de la session](#) vers laquelle il a été traversé.

### 7.2.6.3 L' [PageTransitionEvent](#) interface

✓

MDN

[Exposed=Window]

```
interface PageTransitionEvent : Event {
```

```
  constructor(DOMString type, optional PageTransitionEventInit
```

```
  eventInitDict = {});
```

```
  readonly attribute boolean persisted;
```

```
};
```

```
dictionary PageTransitionEventInit : EventInit {
```

```
  boolean persisted = false;
```



```
};
```

#### **event.persisted**

✓

Pour l' `pageshow` événement, renvoie false si la page vient d'être chargée (et l' `load` événement se déclenchera). Sinon, renvoie vrai.

Pour l' `pagehide` événement, renvoie false si la page disparaît pour la dernière fois. Sinon, renvoie true, ce qui signifie que la page peut être réutilisée si l'utilisateur revient sur cette page (si l' `Document` état `recupérable` de reste true).

Les éléments qui peuvent rendre la page irrécupérable incluent :

- L'agent utilisateur a décidé de ne pas conserver la `Document` vie dans une `entrée d'historique de session` après `le téléchargement`
- Avoir `iframe` des s qui ne sont pas `recupérables`
- `WebSocket` Objets actifs
- `Abandon d'un Document`

L' `persisted` attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente les informations contextuelles de l'événement.

Pour **déclencher un événement de transition de page** nommé `eventName` dans une `Window` fenêtre avec un booléen `persisted` , `déclenchez un événement` nommé `eventName` dans `window` , en utilisant `PageTransitionEvent`, avec l' `persisted` attribut initialisé à `persisted` , l' `cancelable` attribut initialisé à true, l' `bubbles` attribut initialisé à true et l' `indicateur de remplacement de cible` hérité ensemble.

Les valeurs de `cancelable` et `bubbles` n'ont aucun sens, car l'annulation de l'événement ne fait rien et il n'est pas possible de dépasser l' `Window` objet. Ils sont définis sur true pour des raisons historiques.

#### **7.2.6.4 L' `BeforeUnloadEvent` interface**

✓

MDN

```
[Exposed=Window]
```

```
interface BeforeUnloadEvent : Event {
```

```
    attribute DOMString returnValue;
```

```
};
```

Il n'y a pas `BeforeUnloadEvent` de méthodes d'initialisation spécifiques.

L' `BeforeUnloadEvent` interface est une interface héritée qui permet de [vérifier si le déchargement est annulé par l'utilisateur](#) pour être contrôlé non seulement en annulant l'événement, mais en définissant l' `returnValue` attribut sur une valeur en plus de la chaîne vide. Les auteurs doivent utiliser la `preventDefault()` méthode, ou d'autres moyens d'annulation d'événements, au lieu d'utiliser `returnValue`.

L' `returnValue` attribut contrôle le processus de [vérification si le déchargement est annulé par l'utilisateur](#). Lorsque l'événement est créé, l'attribut doit être défini sur la chaîne vide. Lors de l'obtention, il doit renvoyer la dernière valeur à laquelle il a été défini. Lors de la définition, l'attribut doit être défini sur la nouvelle valeur.

*Cet attribut est `DOMString` uniquement pour des raisons historiques. Toute valeur autre que la chaîne vide sera traitée comme une demande de confirmation de l'utilisateur.*

## 7.3 Infrastructure pour les séquences de documents

Cette norme contient plusieurs concepts connexes pour regrouper des séquences de documents. En bref, résumé non normatif :

- [Les éléments navigables](#) sont une représentation visible par l'utilisateur d'une séquence de documents, c'est-à-dire qu'ils représentent quelque chose qui peut être parcouru entre les documents. Des exemples typiques sont des onglets ou des fenêtres dans un navigateur Web, ou [iframes](#), ou [frames](#) dans un fichier [frameset](#).
- [Les navigables traversables](#) sont un type spécial de navigables qui contrôlent l'historique de session d'eux-mêmes et de leurs navigables descendants. C'est-à-dire qu'en plus de leur propre série de documents, ils représentent un arbre d'autres séries de documents, plus la possibilité de parcourir linéairement en arrière et en avant une vue aplatie de cet arbre.
- [Les contextes de navigation](#) sont une représentation destinée aux développeurs d'une série de documents. Ils correspondent 1:1 avec [WindowProxy](#) des objets. Chaque navigable peut présenter une série de contextes de navigation, avec [des basculements](#) entre ces contextes de navigation se produisant dans certaines circonstances bien définies.

La majeure partie de cette norme fonctionne dans le langage des navigables, mais certaines API exposent l'existence de changements de contexte de navigation, et donc certaines parties de la norme doivent fonctionner en termes de contextes de navigation.

### 7.3.1 Navigables

Un **navigable** présente un [Document](#) à l'utilisateur via son [entrée d'historique de session active](#) . Chaque navigable possède :

- Un **identifiant** , une [nouvelle valeur interne unique](#) .
- Un **parent** , un [navigable](#) ou null.
- Une **entrée d'historique de session en cours** , une [entrée d'historique de session](#) .

Cela ne peut être modifié que dans la [file d'attente de parcours de l'historique de session](#) du parent [traversable navigable](#) .

- Une **entrée d'historique de session active** , une [entrée d'historique de session](#) .

Cela ne peut être modifié qu'à partir de la boucle d'événements du document [de l'entrée active de l'historique de la session](#) .

- An **est un booléen fermant** , initialement faux.

*Ceci n'est défini sur true que pour [les navigables traversables de niveau supérieur](#) .*

- An **est un booléen retardant** **load les événements** , initialement faux.

*Ceci n'est défini sur vrai que dans les cas où le [parent](#) de l'élément navigable n'est pas nul.*

L' [entrée de l'historique de la session en cours](#) et l' [entrée de l'historique de la session active](#) sont généralement identiques, mais elles ne sont pas synchronisées lorsque :

- Des navigations synchrones sont effectuées. Ainsi, l' [entrée active de l'historique de la session](#) devance temporairement l' [entrée actuelle de l'historique de la session](#) .
- Une réponse non affichable et sans erreur est reçue pendant [la traversée de l'historique](#) . Cela met à jour l' [entrée d'historique de session actuelle](#) mais laisse l' [entrée d'historique de session active](#) telle quelle.

---

Le **document actif** d'un [élément navigable](#) est [le document](#) de son [entrée d'historique de session active](#) .

*Cela peut être lu en toute sécurité à partir de la [file d'attente de traversée de l'historique de la session](#) du [traversable de niveau supérieur](#) du navigable . Bien que l'[entrée d'historique active](#) d' un [élément navigable](#) puisse changer de manière synchrone, la nouvelle entrée aura toujours la même valeur .[Document](#)*

Le **contexte de navigation actif** d' [un élément navigable](#) est [le contexte de navigation](#) de son [document actif](#) . Si ce [navigable](#) est un [navigable traversable](#) , alors son [contexte de navigation actif](#) sera un [contexte de navigation de niveau supérieur](#) .

L' **actif** d' [un navigable](#) est son [contexte de navigation actif](#) associé .[WindowProxy](#)[WindowProxy](#)

La **fenêtre active** d' [un navigable](#) est sa [ [\[Window\]](#) active ] .[WindowProxy](#)

*Cela sera toujours égal à l'objet global pertinent du [document](#) navigable actif ; ceci est maintenu en synchronisation par l' algorithme [make active](#) .*

Le **nom cible** d'un [élément navigable](#) est [le nom cible navigable](#) de l'état du [document de son entrée active dans l'historique de la session](#) .

---

Pour obtenir le **nœud navigable** d'un *nœud* [nœud](#) , retournez le [navigable](#) dont [le document actif](#) est [le document](#) *nœud* du nœud , ou null s'il n'y a pas de tel [navigable](#) .

---

Pour **initialiser le navigable** [navigable](#) *navigable* , étant donné un [état de document](#) *documentState* et un parent [navigable](#) -or-null facultatif (null par défaut) :

1. Soit *entry* une nouvelle [entrée d'historique de session](#) , avec

[URL](#)  
[URL](#) du *document*  
[état des documents](#)  
*documentState*

*L'appelant de cet algorithme est responsable de l'initialisation de l' [étape](#) d'entrée ; il sera laissé comme " " jusqu'à ce que ce soit terminé.*`pending`

2. Définit [l'entrée de l'historique de la session en cours](#) de *navigable* sur *entry* .
3. Définissez [l'entrée de l'historique des sessions actives](#) de *navigable* sur *entry* .
4. Définissez [le parent](#) de *navigable* sur *parent* .

### 7.3.1.1 Navigables traversables

Un **navigable traversable** est un [navigable](#) qui contrôle également quelle [entrée d'historique de session](#) doit être l' [entrée d'historique de session actuelle](#) et [l'entrée d'historique de session active](#) pour lui-même et ses [navigables](#) descendants .

En plus des propriétés d'un [navigable](#) , un [navigable traversable](#) possède :

- Une **étape de l'historique de la session en cours** , un nombre, initialement 0.

- **Entrées d'historique de session** , une [liste](#) d' [entrées d'historique de session](#) , initialement une nouvelle [liste](#) .
- Une **file d'attente de parcours d'historique de session** , une [file d'attente parallèle de parcours d'historique de session](#) , le résultat du [démarrage d'une nouvelle file d'attente parallèle de parcours d'historique de session](#) .
- Un booléen **d'étape d'historique d'application imbriqué en cours d'exécution** , initialement faux.
- Un **état de visibilité du système** , qui est " `hidden` " ou " `visible` ".

Consultez la section [sur la visibilité de la page](#) pour connaître les exigences relatives à cet élément.

Pour obtenir le **navigable traversable** d'un [navigable](#) *inputNavigable* :

1. Soit *navigable* être *inputNavigable* .
2. Bien que *navigable* ne soit pas un [navigable traversable](#) , définissez *navigable* sur le [parent](#) de *navigable* .
3. Retour *navigable* .

### 7.3.1.2 Traversables de niveau supérieur

Un **traversable de niveau supérieur** est un [navigable traversable](#) avec un [parent](#) nul .

*Actuellement, tous [les navigables traversables](#) sont [des traversables de niveau supérieur](#) . Les propositions futures envisagent d'introduire des traversées non de niveau supérieur.*

Un agent utilisateur contient un **ensemble traversable de niveau supérieur** (un [ensemble](#) de [traversables de niveau supérieur](#) ). Ceux-ci sont généralement présentés à l'utilisateur sous la forme de fenêtres de navigateur ou d'onglets de navigateur.

Pour obtenir le **traversable de niveau supérieur** d'un *inputNavigable* [navigable](#) :

1. Soit *navigable* être *inputNavigable* .
2. Alors que [le parent](#) de *navigable* n'est pas nul, définissez *navigable* sur le [parent](#) de *navigable* .
3. Retour *navigable* .

Pour **créer un nouveau traversable de niveau supérieur** en fonction d'un [contexte de navigation](#) -ou-null *opener* et d'une chaîne *targetName* :

1. Soit *document* nul.
2. Si *opener* est null, définissez *document* sur la deuxième valeur de retour de [la création d'un nouveau contexte de navigation de niveau supérieur et document](#) .
3. Sinon, définissez *document* sur la deuxième valeur de retour de [la création d'un nouveau contexte de navigation auxiliaire et document](#) donné *opener* .
4. Soit *documentState* un nouvel [état de document](#) , avec  
[document](#)  
*document*  
[nom de la cible navigable](#)  
*nom\_cible*
5. Soit *traversable* un nouveau [traversable navigable](#) .
6. [Initialiser le navigable](#) *traversable* donné *documentState* .
7. Soit *initialHistoryEntry* l' [entrée active de l'historique de la](#) session *traversable* .
8. Définissez le [pas](#) de *initialHistoryEntry* sur 0.
9. [Ajouter](#) *initialHistoryEntry* aux [entrées d'historique](#) de session de *traversable* .
10. Si *opener* n'est pas nul, [clonez par héritage un hangar de stockage traversable](#) en fonction des paramètres de [niveau supérieur traversable](#) et *traversable* de *opener* . [\[STOCKAGE\]](#)
11. [Ajoute](#) *traversable* à [l'ensemble traversable de niveau supérieur](#) de l'agent utilisateur .
12. Retour *traversable* .

Pour **créer un nouveau traversable de niveau supérieur** avec une [URL](#) *initialNavigationURL* et une [ressource POST](#) facultative -ou-null *initialNavigationPostResource* (null par défaut) :

1. Soit *traversable* le résultat de [la création d'un nouveau traversable de niveau supérieur](#) donné null et la chaîne vide.
2. [Naviguez](#) *traversable* vers *initialNavigationURL* en utilisant [le document actif](#) de *traversable* , avec [documentResource](#) défini sur *initialNavigationPostResource* .

*Nous traitons ces navigations initiales comme une navigation traversable elle-même, ce qui garantira la réussite de tous les contrôles de sécurité pertinents.*

### 3. Retour *traversable* .

#### 7.3.1.3 Navigables enfants

Certains éléments (par exemple, [iframe](#) des éléments) peuvent présenter une [navigable](#) à l'utilisateur. Ces éléments sont appelés **conteneurs navigables** .

Chaque [conteneur navigable](#) a un **contenu navigable** , qui est soit [navigable](#) soit null. Il est initialement nul.

Le **conteneur** d'un [navigable](#) *navigable* est le [conteneur navigable](#) dont [le contenu navigable](#) est *navigable* , ou null s'il n'y a pas un tel élément.

Le **document conteneur** d'un [navigable](#) *navigable* est le résultat de l'exécution de ces étapes :

1. Si [le conteneur](#) de *navigable* est null, renvoie null.
2. Renvoie le document de [noeud](#) du [conteneur](#) *navigable* .

*Ceci est égal à la [racine shadow-inclusive](#) du [conteneur](#) de *navigable* , car le [conteneur](#) de *navigable* doit être [connecté](#) .*

Le **document conteneur** d'un [Document](#) *document* est le résultat de l'exécution de ces étapes :

1. Si [le noeud navigable](#) du *document* est null, alors retournez null.
2. Renvoie le [document conteneur](#) du [noeud navigable](#) du document .

Un [navigable](#) *navigable* est un **enfant navigable** d'un autre *navigable potentialParent* lorsque [le parent](#) de *navigable* est *potentialParent* . On peut aussi simplement dire qu'un [navigable](#) "est un [enfant navigable](#) ", ce qui signifie que son [parent](#) est non nul.

*Tous [les éléments navigables enfants](#) sont le [contenu navigable](#) de leur [conteneur](#) .*

Le **document de contenu** d'un *conteneur* [de conteneur navigable](#) est le résultat de l'exécution de ces étapes :

1. Si [le contenu navigable](#) du *conteneur* est null, alors retournez null.
2. Soit *document* le [document actif](#) du [contenu navigable](#) du *conteneur* .
3. Si [l'origine](#) du *document* et [l'origine](#) du [document du noeud](#) du *conteneur* ne sont pas [identiques origin-domain](#) , alors retournez null.



#### 4. Document de retour .

La **fenêtre de contenu** d'un *conteneur* [navigable](#) est le résultat de l'exécution de ces étapes :

1. Si [le contenu navigable](#) du *conteneur* est null, alors retournez null.
  2. Renvoie l'objet actif du [contenu navigable](#) du *conteneur* `.WindowProxy`
- 

Pour **créer un nouvel enfant navigable** , étant donné un *élément* `element` :

1. Soit *parentNavigable* le [nœud navigable](#) de l'*élément* .
2. Soit *group* le nœud de l' *élément* le [groupe](#) du contexte de [navigation](#) de [niveau supérieur](#) du [document](#) .
3. Laissons *browserContext* et *document* être le résultat de [la création d'un nouveau contexte de navigation et d'un document](#) donné par le nœud [document](#) , *element* et *group* de l'*élément* .
4. Laissez *targetName* être nul.
5. Si l'*élément* a un `name`attribut de contenu, définissez *targetName* sur la valeur de cet attribut.
6. Soit *documentState* un nouvel [état de document](#) , avec  
[document](#)  
*document*  
[nom de la cible navigable](#)  
*nom\_cible*
7. Soit *navigable* un nouveau [navigable](#) .
8. [Initialisez le navigable](#) *navigable* donné *documentState* et *parentNavigable* .
9. Définissez le contenu de l' *élément* [navigable](#) sur *navigable* .
10. Soit *historyEntry* l' [entrée d'historique de session active](#) de *navigable* .
11. Soit *traversable* le [navigable traversable](#) de *parentNavigable* .
12. [Ajoutez les étapes de parcours d'historique de session suivantes](#) à *traversable* :
  1. Soit *parentDocState* l' état du [document de l'entrée d'historique de session active](#) de *parentNavigable* .

2. Soit *targetStepSHE* la première [entrée d'historique de session](#) dans [les entrées d'historique](#) de session de *traversable* dont [l'état du document](#) est égal à *parentDocState* .
3. Définissez le [pas](#) de *historyEntry* sur le [pas](#) de *targetStepSHE* .
4. Soit *nestedHistory* un nouvel [historique imbriqué](#) dont [l'id](#) est l' [id](#) de *navigable* et [la liste des entrées](#) est « *historyEntry* ».
5. [Ajouter](#) *nestedHistory* aux [historiques imbriqués](#) de *parentDocState* .
6. [Appliquez les modifications d'historique en attente](#) à *traversable* .

### 7.3.1.4 Diagrammes de Jake

Une méthode utile pour visualiser les séquences de documents, et en particulier [les navigables](#) et leurs [entrées d'historique de session](#) , est le **diagramme de Jake** . Un diagramme de Jake typique est le suivant :

	0	1	2	3	4
top	/ta			:/ta#foo	/to
frames[0]	/i-0-a	/i-0-b			
frames[1]	/i-1-a		/i-1-b		

Ici, chaque colonne numérotée indique une valeur possible pour l' [étape de l'historique de session](#) du traversable . Chaque ligne étiquetée représente un [navigable](#) , lors de la transition entre différentes URL et documents. Le premier, étiqueté *top*, étant le [plus haut niveau traversable](#) , et les autres étant [des enfants navigables](#) . Les documents sont donnés par la couleur de fond de chaque cellule, avec une nouvelle couleur de fond indiquant un nouveau document dans ce fichier [navigable](#) . Les URL sont données par le contenu textuel des cellules ; généralement, ils sont donnés sous forme [d'URL relatives](#) par souci de brièveté, à moins qu'un cas d'origine croisée ne fasse spécifiquement l'objet d'une enquête. Un navigable donné peut ne pas exister à une étape donnée, auquel cas les cellules correspondantes sont vides. Le numéro d'étape en gras et en italique représente l' [étape d'historique de session en cours](#) du traversable, et toutes les cellules avec des URL en gras et en italique représentent l' [entrée d'historique de session en cours](#) pour le navigable de cette ligne.

Ainsi, le diagramme de Jake ci-dessus décrit la séquence d'événements suivante :

1. Un [traversable de niveau supérieur](#) est créé, commençant par l' URL */t-a*, avec deux [navigables enfants](#) commençant respectivement par */i-0-a* et */i-1-a*

2. Le premier enfant navigable est [dirigé](#) vers un autre document, avec URL `/i-0-b`.
3. Le deuxième enfant navigable est [dirigé](#) vers un autre document, avec URL `/i-1-b`.
4. Le traversable de niveau supérieur est [dirigé](#) vers le *même* document, mettant à jour son URL en `/t-a#foo`.
5. Le traversable de niveau supérieur est [dirigé](#) vers un autre document, avec URL `/t-b`. (Remarquez comment ce document, bien sûr, ne reprend pas les navigables enfants de l'ancien document.)
6. Le traversable a été [traversé par un delta](#) de -3, retour à l'étape 1.

[Les diagrammes de Jake](#) sont un outil puissant pour visualiser les interactions de plusieurs navigables, navigations et traversées. Ils ne peuvent pas capturer toutes les interactions possibles — par exemple, ils ne fonctionnent qu'avec un seul niveau d'imbrication — mais nous aurons l'occasion de les utiliser pour illustrer plusieurs situations complexes tout au long de cette norme.

[Les diagrammes de Jake](#) portent le nom de leur créateur, l'inimitable Jake Archibald.

### 7.3.1.5 Collections navigables associées

Il est souvent utile dans les algorithmes de cette norme de regarder des collections de [navigables](#) commençant à un [Document](#). Cette section contient un ensemble organisé d'algorithmes pour collecter ces éléments navigables.

*Les valeurs de retour de ces algorithmes sont ordonnées de sorte que les parents apparaissent avant leurs enfants. Les appelants comptent sur cette commande. Commencer par un [Document](#), plutôt que par un [navigable](#), est généralement préférable car cela permet à l'appelant de savoir s'il commence par un [entièrement actif Document](#) ou non. Bien que les [s non pleinement actifs Document](#) aient des navigables ancêtre et descendant, ils se comportent souvent comme s'ils n'en avaient pas (par exemple, dans le `window.parent.getter`).*

Les **navigables ancêtres** d'un [Document](#) *document* sont donnés par ces étapes :

1. Soit *navigable* le [nœud](#) parent du *document* [navigable](#) .
2. Soit *les ancêtres* une liste vide.
3. Tandis que *navigable* n'est pas nul :
  1. [Préfixe](#) *navigable* jusqu'aux *ancêtres* .
  2. Définissez *navigable* sur le [parent](#) de *navigable* .

4. Retour *des ancêtres* .

Les **navigables ancêtres inclusifs** d'un Document *document* sont donnés par ces étapes :

1. Soit *navigables* les navigables ancêtres du *document* .
2. Ajouter le nœud navigable du *document* aux *navigables* .
3. Retour *navigables* .

Les **navigables descendantes** d'un Document *document* sont données par ces étapes :

1. Soit *navigables* une nouvelle liste .
2. Soit *navigableContainers* une liste de tous les descendants shadow-inclusives de *document* qui sont des conteneurs navigables , dans l'ordre de l'arborescence shadow-inclusive .
3. Pour chaque *navigableContainer* de *navigableContainer* :
  1. Si le contenu *navigableContainer* de *navigableContainer* est nul, continuez.
  2. Étendez les *éléments navigables* avec les éléments *navigables* descendants inclusifs du document actif *navigable* du contenu de *navigableContainer* .
4. Retour *navigables* .

Les **navigables descendants inclusifs** d'un Document *document* sont donnés par ces étapes :

1. Soit *navigables* le « nœud navigable du *document* ».
2. Étendre les *navigables* avec les navigables descendants du *document* .
3. Retour *navigables* .

*Ces algorithmes de collecte de descendants sont décrits comme examinant l'arborescence DOM des Document objets descendants. En réalité, cela n'est souvent pas faisable puisque l'arbre DOM peut être dans un autre processus de l'appelant de l'algorithme. Au lieu de cela, les implémentations répliquent généralement les arborescences appropriées à travers les processus.*

Les **éléments navigables enfants de l'arborescence de documents** d'un Document *document* sont donnés par ces étapes :

1. Si le nœud navigable du *document* est nul, renvoie la liste vide.

2. Soit *navigables* une nouvelle [liste](#) .
3. Soit *navigableContainers* une [liste](#) de tous [les descendants](#) de *document* qui sont [des conteneurs navigables](#) , dans [l'ordre de l'arborescence](#) .
4. [Pour chaque](#) *navigableContainer* de *navigableContainer* :
  1. Si [le contenu navigable](#) de *navigableContainer* est null, [continuez](#) .
  2. [Ajouter](#) le contenu de *navigableContainer* [navigable](#) à *navigables* .
5. Retour *navigables* .

### 7.3.1.6 Destruction navigable

Pour **détruire un enfant navigable** étant donné un *container* [container navigable](#) :

1. Soit *navigable* le [contenu navigable](#) du *conteneur* .
2. Si *navigable* est null, alors retournez.
3. Définissez [le contenu navigable](#) du *conteneur* sur null.
4. [Détruire le document actif](#) de *navigable* .
5. Soit *parentDocState* l' état du [document de l'entrée d'historique de session active](#) du [nœud navigable](#) du *conteneur* .
6. [Supprimez](#) l' [historique imbriqué](#) des [historiques imbriqués](#) de *parentDocState* dont [l'identifiant](#) est égal à [l'identifiant](#) de *navigable* .
7. Soit *traversable* le [nœud navigable](#) du *conteneur* navigable [traversable navigable](#) .
8. [Ajoutez les étapes de parcours d'historique de session suivantes](#) à *traversable* :
  1. [Appliquez les modifications d'historique en attente](#) à *traversable* .

Pour **détruire un traversable de niveau supérieur** étant donné un *traversable* [traversable de niveau supérieur](#) :

1. Soit *browserContext* le [contexte de navigation actif](#) de *traversable* .
2. [Pour chaque](#) *historyEntry* dans [les entrées d'historique de session](#) de *traversable* dans quel ordre ? :
  1. Soit *document* le [document](#) de *historyEntry* .

2. Si *document* n'est pas nul, alors [détruisez](#) *document* .
3. [Supprimez](#) *browserContext* .
4. Supprimer *traversable* de l'interface utilisateur (par exemple, fermer ou masquer son onglet dans un navigateur à onglets).
5. [Supprimez](#) *traversable* de [l'ensemble traversable de niveau supérieur](#) de l'agent utilisateur .

Les agents utilisateurs peuvent [détruire un traversable de niveau supérieur](#) à tout moment (généralement, [en réponse aux requêtes des utilisateurs](#) ).

Pour **fermer un traversable de niveau supérieur** *traversable* :

1. Soit *toUnload* les [navigables descendants inclusifs](#) du document [actif](#) *traversable* .
2. Si le résultat de [la vérification si le déchargement est annulé par l'utilisateur](#) pour *toUnload* est vrai, alors retournez.
3. [Déchargez](#) les [documents actifs](#) de chacun de *toUnload* . Dans quel ordre?
4. [Détruire](#) *traversable* .

### 7.3.1.7 Noms de cibles navigables

[Les éléments navigables](#) peuvent recevoir [des noms de cible](#) , qui sont des chaînes permettant à certaines API (telles que `window.open()` ou l'attribut `a` de l'élément `target`) de cibler [les navigations](#) sur cet élément navigable.

Un **nom de cible navigable valide** est une chaîne contenant au moins un caractère qui ne commence pas par un caractère U+005F LOW LINE. (Les noms commençant par un trait de soulignement sont réservés à des mots-clés spéciaux.)

Un **nom de cible navigable ou un mot-clé valide** est une chaîne qui est soit un [nom de cible navigable valide](#) , soit une correspondance [ASCII non sensible à la casse](#) pour l'un des éléments suivants : `_blank`, `_self`, `_parent`, ou `_top`.

Ces valeurs ont des significations différentes selon que la page est en bac à sable ou non, comme résumé dans le tableau (non normatif) suivant. Dans ce tableau, "actuel" signifie le [navigable](#) dans lequel se trouve le lien ou le script, "parent" signifie le [parent](#) du [navigable](#) dans lequel se trouve le lien ou le script, "top" signifie le [traversable de niveau supérieur](#) du [navigable](#) dans lequel se trouve le lien ou le script est dans, "nouveau" signifie un nouveau [navigable traversable](#) avec un [parent](#) nul (qui peut utiliser un [contexte de navigation auxiliaire](#), sous réserve de diverses préférences utilisateur et politiques d'agent utilisateur), "aucun" signifie que

rien ne se passera, et "peut-être nouveau" signifie la même chose que "nouveau" si le mot-clé " " est également spécifié sur l'attribut (ou si l'utilisateur [allows popups](#) a [sandbox](#) remplacé le sandboxing), et identique à "aucun" sinon.

Mot-clé	Effet ordinaire	Effet dans un <a href="#">iframe</a> avec...	
		<code>sandbox=""</code>	<code>sandbox="allow-top-navigation"</code>
aucun spécifié, pour les liens et les soumissions de formulaires	actuel	actuel	actuel
chaîne vide	actuel	actuel	actuel
<code>_blank</code>	nouveau	peut-être nouveau	peut-être nouveau
<code>_self</code>	actuel	actuel	actuel
<code>_parent</code> s'il n'y a pas de parent	actuel	actuel	actuel
<code>_parent</code> si le parent est aussi en haut	parent/supérieur	aucun	parent/supérieur
<code>_parent</code> s'il y en a un et c'est pas top	parent	aucun	aucun
<code>_top</code> si top est courant	actuel	actuel	actuel
<code>_top</code> si top n'est pas actuel	haut	aucun	haut
nom qui n'existe pas	nouveau	peut-être nouveau	peut-être nouveau
nom qui existe et est un descendant	descendant spécifié	descendant spécifié	descendant spécifié
nom existant et actuel	actuel	actuel	actuel
nom qui existe et est un ancêtre qui est top	ancêtre spécifié	aucun	ancêtre/sommet spécifié
nom qui existe et qui est un ancêtre qui n'est pas top	ancêtre spécifié	aucun	aucun
autre nom qui existe avec un sommet commun	spécifié	aucun	aucun
nom qui existe avec un top différent, s'il est <a href="#">familier</a> et <a href="#">un navigateur sandbox autorisé</a>	spécifié	spécifié	spécifié
nom qui existe avec un top différent, s'il est <a href="#">familier</a> mais pas <a href="#">un navigateur sandbox autorisé</a>	spécifié	aucun	aucun
nom qui existe avec un top différent, pas <a href="#">familier</a>	nouveau	peut-être nouveau	peut-être nouveau



La plupart des restrictions sur les contextes de navigation en bac à sable sont appliquées par d'autres algorithmes, par exemple l' algorithme [de navigation](#) , et non les [règles de choix d'un navigable](#) données ci-dessous.

---

**Les règles pour choisir un navigable** , étant donné un *nom* de chaîne , un [navigable](#) *currentNavigable* et un booléen *noopener* sont les suivantes :

1. Soit *choisi* nul.
2. Laissez *windowType* être " `existing or none`".
3. Soit *sandboxingFlagSet* le [jeu d'indicateurs de sandboxing](#) actif du [document actif](#) de *currentNavigable* .
4. Si *name* est la chaîne vide ou une correspondance [ASCII non sensible à la casse](#) pour " `_self`", alors définissez *choisi* sur *currentNavigable* .
5. Sinon, si *name* est une correspondance [ASCII insensible à la casse](#) pour " `_parent`", définissez le parent [de](#) *currentNavigable* , le cas échéant, et *currentNavigable* dans le cas contraire.
6. Sinon, si *name* est une correspondance [ASCII insensible à la casse](#) pour " `_top`", définissez l'élément *choisi* sur le [traversable navigable](#) de *currentNavigable* .
7. Sinon, si *name* n'est pas une correspondance [ASCII insensible à la casse](#) pour " `_blank`", il existe un [navigable](#) dont [le nom cible](#) est le même que *name* , [le contexte de navigation actif](#) de *currentNavigable* est [familier avec le contexte de navigation actif](#) de ce [navigable](#) et l'agent utilisateur détermine que les deux contextes de navigation sont suffisamment liés pour qu'il soit correct s'ils s'atteignent, ensemble *choisi* pour navigable. S'il existe plusieurs éléments [navigables correspondants](#), l'agent utilisateur doit en choisir un d'une manière arbitrairement cohérente, comme le plus récemment ouvert, le plus récemment ciblé ou le plus étroitement lié, et l'ensemble *choisi* pour lui.

Cela sera précisé dans [le numéro 313](#) .

8. Sinon, un nouveau [traversable de niveau supérieur](#) est demandé, et ce qui se passe dépend de la configuration et des capacités de l'agent utilisateur — il est déterminé par les règles données pour la première option applicable dans la liste suivante :

**Si [la fenêtre active](#) de *currentNavigable* n'a pas [d'activation transitoire](#) et que l'agent utilisateur a été configuré pour ne pas afficher les fenêtres**



**contextuelles (c'est-à-dire que l'agent utilisateur a un "bloqueur de fenêtres contextuelles" activé)**

L'agent utilisateur peut informer l'utilisateur qu'une fenêtre contextuelle a été bloquée.

**Si *sandboxingFlagSet* a l'indicateur [de contexte de navigation de navigation auxiliaire en bac à sable défini](#)**

L'agent utilisateur peut signaler à une console de développeur qu'une fenêtre contextuelle a été bloquée.

**Si l'agent utilisateur a été configuré de telle sorte que dans cette instance, il créera un nouveau [parcours de niveau supérieur](#)**

1. Définissez *windowType* sur " `new and unrestricted`".
2. Soit *currentDocument* le [document actif](#) de *currentNavigable* .
3. Si [la valeur](#) de [la stratégie d'ouverture croisée d'origine](#) de *currentDocument* est " " ou " ", et que [l'origine](#) de *currentDocument* n'est pas [la même origine](#) que [l'origine de niveau supérieur](#) de [l'objet de paramètres pertinents](#) de *currentDocument* , alors :[same-originsame-origin-plus-COEP](#)
  1. Définissez *noopener* sur `true`.
  2. Définissez *le nom* sur " `_blank`".
  3. Définissez *windowType* sur " `new with no opener`".

*En présence d'une [stratégie d'ouverture d'origine croisée](#) , les documents imbriqués qui sont d'origine croisée avec le document actif de leur contexte de navigation de niveau supérieur définissent toujours *noopener* sur `true`.*

4. Soit *choisi* nul.
5. Soit *targetName* la chaîne vide.
6. Si *name* n'est pas une correspondance [ASCII insensible à la casse](#) pour " `_blank`", définissez *targetName* sur *name* .
7. Si *noopener* est vrai, alors définissez *choisi* sur le résultat de [la création d'un nouveau traversable de niveau supérieur](#) donné null et *targetName* .
8. Sinon:
  1. Définissez le résultat de la création d'un nouveau traversable de [niveau supérieur](#) en fonction [du contexte de navigation actif](#) de *currentNavigable* et de *targetName* .

2. Si l'[indicateur de contexte de navigation de navigation en bac à sable](#) de `sandboxingFlagSet` est défini, définissez le seul [navigateur en bac à sable autorisé](#) du [contexte de navigation actif](#) choisi sur [le contexte de navigation actif](#) de `currentNavigable`.
9. Si l'indicateur `sandboxingFlagSet` [se propage vers les contextes de navigation auxiliaires](#) est défini, tous les indicateurs définis dans `sandboxingFlagSet` doivent être définis dans [l'indicateur de sandboxing](#) contextuel du contexte [de navigation actif](#) choisi.

*Si le [navigable nouvellement créé](#) choisi est immédiatement [navigué](#), alors la navigation se fera avec "[replace](#)" [le comportement de gestion de l'historique](#).*

**Si l'agent utilisateur a été configuré de telle sorte que dans cette instance, il choisira `currentNavigable`**

Définissez `choisi` sur `currentNavigable`.

**Si l'agent utilisateur a été configuré de sorte que dans ce cas, il ne trouvera pas de**

Ne fais rien.

*Les agents utilisateurs sont encouragés à fournir aux utilisateurs un moyen de configurer l'agent utilisateur pour qu'il choisisse toujours `currentNavigable`.*

9. Renvoie `choisi` et `windowType`.

### 7.3.2 Contextes de navigation

Un **contexte de navigation** est une représentation programmatique d'une série de documents, dont plusieurs peuvent vivre dans un même [fichier navigable](#). A chaque [contexte de navigation](#) correspond un `WindowProxy` objet, ainsi que :

- Un **contexte de navigation d'ouverture**, un [contexte de navigation](#) ou nul, initialement nul.
- Un **opener origin à creation**, un [origin](#) or null, initialement null.
- An **est** un booléen contextuel, initialement faux.

*Le seul impact obligatoire dans cette spécification de [popup](#) est sur le [visible](#) getter des `BarProp` objets concernés. Cependant, les agents utilisateurs peuvent également l'utiliser pour [des considérations d'interface utilisateur](#).*

- An **est** un booléen auxiliaire, initialement faux.

- Une **URL initiale** , une [URL](#) ou null, initialement null.
- Une **URL de base du créateur** , null ou un algorithme qui renvoie une [URL](#) , initialement null.
- Un entier **d'ID de groupe de contexte de navigation virtuelle** , initialement 0. Il est utilisé par [les rapports de politique d'ouverture cross-origin](#) , pour garder une trace des changements de groupe de contexte de navigation qui se seraient produits si la politique de rapport uniquement avait été appliquée.

La **fenêtre active** d' un [contexte de navigation](#) est la valeur d'emplacement interne `[[Window]]` de son objet . Le **document actif** d' un [contexte de navigation](#) est celui [associé](#) à sa fenêtre [active](#) `.WindowProxyDocument`

Le **traversable de niveau supérieur** d' un [contexte de navigation](#) est [le traversable de niveau supérieur](#) du [nœud navigable](#) de son [document actif](#) .

Un [contexte de navigation](#) dont [est auxiliaire](#) est vrai est appelé **contexte de navigation auxiliaire** . Les contextes de navigation auxiliaires sont toujours [des contextes de navigation de niveau supérieur](#) .

Il n'est pas clair si un concept distinct [est auxiliaire](#) est nécessaire. Dans [le numéro 5680](#) , il est indiqué que nous pouvons simplifier cela en utilisant si le [contexte de navigation de l'ouvreur](#) est nul ou non.

*Les spécifications modernes devraient éviter d'utiliser le concept [de contexte de navigation](#) dans la plupart des cas, à moins qu'elles ne traitent des subtilités des [changements de groupe de contexte de navigation](#) et de [l'allocation des clusters d'agents](#) . Au lieu de cela, les concepts [Document](#) et [navigable](#) sont généralement plus appropriés.*

---

**Document** Le **contexte de navigation d'un** est un [contexte de navigation](#) ou nul, initialement nul.

A [Document](#) n'a pas nécessairement un [contexte de navigation](#) non nul . En particulier, les outils de data mining sont susceptibles de ne jamais instancier les contextes de navigation. Un [Document](#) créé à l'aide d'une API telle que `never` a un [contexte de navigation](#) `createDocument()` non nul . Et le créé à l'origine pour un élément, qui a depuis été [supprimé du document](#) , n'a pas de contexte de navigation associé, puisque ce contexte de navigation a été [annulé](#) `.Documentiframe` . En général, il existe un mappage 1 à 1 de l' `Window` objet à l' `Document` objet, tant que l' objet a un [contexte de navigation](#) `Document` non nul . Il y a une exception. A peut être réutilisé pour la présentation d'un second dans le même [contexte de navigation](#) , de sorte que le mapping est alors 1-to-2. Cela se produit lorsqu'un [contexte de](#)

*navigation est navigué de l' initiale à l'autre, ce qui sera fait avec remplacement .WindowDocumentabout:blank Document*

### 7.3.2.1 Création de contextes de navigation

Pour **créer un nouveau contexte de navigation et un nouveau document** , donné null ou un *créateur*Document d' objet , null ou un élément *incorporé* , et un groupe de groupe de contexte de navigation :

1. Soit *browserContext* un nouveau contexte de navigation .
2. Soit *unsafeContextCreationTime* l' heure actuelle partagée non sécurisée .
3. Laissez *creatorOrigin* être nul.
4. Si *le créateur* est non nul, alors :
  1. Définissez *creatorOrigin* sur l'origine du *créateur* .
  2. Définissez l' URL de base du créateur de *scanningContext* sur un algorithme qui renvoie l' URL de base du *créateur* .
  3. Définissez l' ID de groupe de contexte de navigation virtuelle de scanningContext sur l' ID de groupe de contexte de navigation virtuelle de contexte de navigation de niveau supérieur du *créateur* .
5. Soit *sandboxFlags* le résultat de la détermination des drapeaux de sandboxing de création donnés *scanningContext* et *embedder* .
6. Soit *origin* le résultat de la détermination de l'origine given about:blank, *sandboxFlags* , *creatorOrigin* et null.
7. Laissez *permissionsPolicy* être le résultat de la création d'une politique d'autorisations en fonction de l'*embedder* et de l'*origine* . [POLITIQUE D'AUTORISATION]
8. Soit *agent* le résultat de l'obtention d'un agent de fenêtre d'origine similaire donné *origin* , *group* et *false* .
9. Supposons que *le contexte d'exécution du domaine* soit le résultat de la création d'un nouvel agent donné par le domaine et des personnalisations suivantes :
  1. Pour l'objet global, créez un nouvel Window objet.
  2. Pour la liaison globale **this** , utilisez l'objet de *scanningContext*WindowProxy .

10. Soit *topLevelCreationURL* si *embedder*[about:blank](#) est null ; sinon, l' [URL de création de niveau supérieur](#) de [l'objet de paramètres pertinents](#) de *l'intégrateur* .
11. Soit *topLevelOrigin* être *origin* si *embedder* est null ; sinon l' [origine de niveau supérieur](#) de l' objet des [paramètres pertinents](#) de *l' intégrateur* .
12. [Configurez un objet de paramètres d'environnement de fenêtre](#) avec [about:blank](#), *realm execution context* , null, *topLevelCreationURL* et *topLevelOrigin* .
13. Soit *loadTimingInfo* une nouvelle [information de synchronisation de chargement de document](#) avec son [heure de début de navigation](#) définie sur le résultat de l'appel [du temps grossier](#) avec *unsafeContextCreationTime* et la [capacité d'isolement cross-origin](#) du nouvel [objet de paramètres d'environnement](#) .
14. Soit *document* un new [Document](#), avec :

[taper](#)

" html "

[type de contenu](#)

" [text/html](#) "

[mode](#)

" quirks "

[origine](#)

*origine*

[contexte de navigation](#)

*contexte de navigation*

[politique d'autorisations](#)

*permissionsPolicy*

[ensemble d'indicateurs de sandboxing actif](#)

*sandboxDrapeaux*

[informations de synchronisation de chargement](#)

*loadTimingInfo*

[est initiale](#)[about:blank](#)

vrai

15. Si *le créateur* est non nul, alors :
  1. Définissez [le référent](#) du *document* sur la [sérialisation](#) de [l'URL](#) du *créateur* .
  2. Définissez [le conteneur de stratégie](#) du *document* sur un [clone](#) du [conteneur de stratégie](#) du *créateur* .
  3. Si [l'origine](#) du *créateur* est [la même origine](#) que l'origine de [niveau supérieur](#) de [l'objet des paramètres pertinents](#) du *créateur* , alors définissez [la stratégie d'ouverture d'origine croisée](#) du *document* sur [le document actif du contexte de navigation](#) de [niveau supérieur du](#)

[contexte de navigation](#) du *créateur* s [politique d'ouverture d'origine croisée](#) .

16. [Assert](#) : l'[URL](#) du *document* et l'[URL de création](#) de l'[objet de paramètres pertinents](#) du *document* sont [.about:blank](#)
17. Marquer le *document* comme [prêt pour les tâches de post-chargement](#) .
18. Assurez-vous que le *document* a un seul [html](#) nœud enfant, qui a lui-même deux nœuds enfants vides : un [head](#) élément et un [body](#) élément.
19. [Rendre](#) le *document* actif .
20. [Terminez complètement le chargement](#) du *document* .
21. Retourne *scanningContext* et *document* .

Pour **créer un nouveau contexte de navigation et un document de niveau supérieur** :

1. Laissez *group* et *document* être le résultat de [la création d'un nouveau contexte de navigation group and document](#) .
2. Renvoie [l'ensemble de contextes de navigation](#) du *groupe* [0] et le *document* .

Pour **créer un nouveau contexte de navigation auxiliaire et un nouveau document** , étant donné un *ouvreur* [de contexte de navigation](#) :

1. Soit *openerTopLevelBrowsingContext* le [contexte de navigation actif](#) du [traversable de niveau supérieur de](#) l' *ouvreur* .
2. Soit *group* le [groupe](#) de *openerTopLevelBrowsingContext* .
3. [Assert](#) : le *groupe* n'est pas nul, car [la navigation](#) l'invoque directement.
4. Définissez *browserContext* et *document* comme le résultat de [la création d'un nouveau contexte de navigation et d'un nouveau document](#) avec *opener* s [active document](#) , null et *group* .
5. Définissez [l'auxiliaire](#) de *scanningContext* sur true.
6. [Ajoutez](#) le *contexte de navigation* au *groupe* .
7. Définissez [le contexte de navigation](#) de l'ouvreur de *scanningContext* sur *opener* .
8. Définissez l' [ID de groupe de contexte de navigation virtuelle](#) de *scanningContext* sur l' [ID de groupe de contexte de navigation virtuelle](#) de *openerTopLevelBrowsingContext* .
9. Définissez l'origine de l'ouvreur de *scanningContext* [à la création](#) sur [l'origine](#) du [document actif](#) de l'ouvreur .

10. Retourne *scanningContext* et *document* .

Pour **déterminer l' origine** , étant donné une [URL](#) *url* , un [indicateur de sandboxing défini](#) *sandboxFlags* , un [origin](#) -or-null *sourceOrigin* et un [origin](#) -or-null *containerOrigin* :

1. Si *sandboxFlags* a son [indicateur de contexte de navigation d'origine en bac à sable](#) défini, renvoie alors un nouvel [opaque origin](#) .
2. Si *url* est null, alors retourne une nouvelle [opaque origin](#) .
3. Si *url* est [about:srcdoc](#), alors :
  1. [Assert](#) : *containerOrigin* est non nul.
  2. Renvoie *containerOrigin* .
4. Si *url* [correspond](#) [about:blank](#) et que *sourceOrigin* n'est pas nul, alors renvoie *sourceOrigin* .
5. [Origine](#) de *url* de retour .

Les cas qui renvoient *sourceOrigin* ou *containerOrigin* donnent deux [Documents](#) qui se retrouvent avec le même [origin](#) sous-jacent , ce qui signifie qu'ils [document.domain](#) affectent les deux.

### 7.3.2.2 Contextes de navigation associés

Un [contexte de navigation](#) *potentielDescendant* est dit ancêtre **d'** un contexte de navigation *potentielAncêtre* si l'algorithme suivant renvoie vrai :

1. Soit *potentialDescendantDocument* le [document actif](#) de *potentialDescendant* .
2. Si *potentialDescendantDocument* n'est pas [entièrement active](#) , alors renvoie false.
3. Soit *ancestorBCs* la liste obtenue en prenant le [contexte de navigation](#) du [document actif](#) de chaque membre des navigables [ancêtres de](#) *potentialDescendantDocument* .
4. Si *ancestorBCs* [contient](#) *potentialAncestor* , alors retourne true.
5. Renvoie faux.

Un **contexte de navigation de niveau supérieur** est un [contexte de navigation](#) dont le [nœud navigable](#) du [document actif](#) est un [navigable traversable](#) .



*Il n'est pas nécessaire qu'il s'agisse d'un fichier traversable de niveau supérieur .*

Le **contexte de navigation de niveau supérieur** d'un début de contexte de navigation est le résultat de l'algorithme suivant :

1. Si le document actif de *start* n'est pas complètement actif , alors renvoie null.
2. Soit *navigable* le nœud navigable du document actif de départ .
3. Alors que le parent de *navigable* n'est pas nul, définissez *navigable* sur le parent de *navigable* .
4. Renvoie le contexte de navigation actif de *navigable* .

*Les termes contexte de navigation ancêtre et contexte de navigation de niveau supérieur sont rarement utiles, car les contextes de navigation en général sont généralement le concept de spécification inapproprié à utiliser . Notez en particulier que lorsqu'un document actif d'un contexte de navigation n'est pas entièrement actif , il ne compte jamais comme un ancêtre ou un contexte de navigation de niveau supérieur, et en tant que tel, ces concepts ne sont pas utiles lorsque bfcache est en jeu.*

*À la place, utilisez des concepts tels que la collection navigables ancêtre , le navigable parent ou le traversable de niveau supérieur d'un navigable .*

---

Un contexte de navigation *A* est **familier avec** un second contexte de navigation *B* si l'algorithme suivant renvoie vrai :

1. Si l'origine du document actif de *A* est la même origine que l'origine du document actif de *B* , alors renvoie vrai.
2. Si le contexte de navigation de niveau supérieur de *A* est *B* , alors retourne true.
3. Si *B* est un contexte de navigation auxiliaire et que *A* est familier avec le contexte de navigation d'ouverture de *B* , alors retourne true.
4. S'il existe un contexte de navigation ancêtre de *B* dont le document actif a la même origine que le document actif de *A* , alors retourne vrai.

*Cela inclut le cas où *A* est un contexte de navigation ancêtre de *B* .*

5. Renvoie faux.



### 7.3.2.3 Regroupements de contextes de navigation

Un [contexte de navigation de niveau supérieur](#) a un **groupe** associé (null ou un [groupe de contexte de navigation](#) ). Il est initialement nul.

Un agent utilisateur détient un **ensemble de groupes de contextes de navigation** (un [ensemble](#) de [groupes de contextes de navigation](#) ).

Un **groupe de contextes de navigation** contient un **ensemble de contextes de navigation** (un [ensemble](#) de [contextes de navigation de niveau supérieur](#) ).

*Un [contexte de navigation de niveau supérieur](#) est ajouté au groupe lors de la création du groupe . Tous les [contextes de navigation de niveau supérieur](#) ajoutés au groupe seront [des contextes de navigation auxiliaires](#) .*

Un [groupe de contexte de navigation](#) a une **carte de cluster d'agents** associée (une [carte](#) faible des [clés de cluster d'agents](#) aux [clusters d'agents](#) ). Les agents utilisateurs sont responsables de la collecte des clusters d'agents lorsqu'il est estimé que plus rien ne peut y accéder.

Un [groupe de contexte de navigation](#) a une **mappe de clés de cluster d'agents historique** associée , qui est une [mappe](#) d' [origines](#) vers [des clés de cluster d'agents](#) . Cette carte est utilisée pour assurer la cohérence de la fonction [de clusters d'agents à clé d'origine](#) en enregistrant les clés de cluster d'agents précédemment utilisées pour une origine donnée.

*La [mappe de clés du cluster d'agents historique](#) n'obtient des entrées que pendant la durée de vie du groupe de contexte de navigation.*

Un [groupe de contexte de navigation](#) a un **mode d'isolation cross-origin** , qui est un [mode d'isolation cross-origin](#) . C'est initialement " [none](#) ".

Un **mode d'isolement d'origine croisée** est l'une des trois valeurs possibles : " [none](#) ", " [logical](#) " ou " [concrete](#) ".

*" [logical](#) " et " [concrete](#) " sont similaires. Ils sont tous deux utilisés pour [parcourir des groupes de contexte](#) où :*

- *chaque niveau supérieur [Document a](#) ``, et [Cross-Origin-Opener-Policy: same-origin](#)*
- *every [Document a un](#) [Cross-Origin-Embedder-Policy](#) en-tête `` dont la valeur est [compatible avec l'isolation cross-origin](#) .*

*Sur certaines plates-formes, il est difficile de fournir les propriétés de sécurité requises pour accorder un accès sécurisé aux API protégées par la [capacité d'isolement cross-origin](#) . Par conséquent, seul " [concrete](#) " peut accorder l'accès à cette fonctionnalité. " [logical](#) " est utilisé sur une plate-forme qui ne prend pas en*

*charge cette capacité, où diverses restrictions imposées par l'isolement entre origines s'appliqueront toujours, mais la capacité n'est pas accordée.*

Pour **créer un nouveau groupe de contexte de navigation et document** :

1. Soit `group` un nouveau [groupe de contexte de navigation](#) .
2. [Ajouter](#) le groupe à [l'ensemble de groupes de contexte de navigation](#) de l'agent utilisateur .
3. Laisser `browserContext` et `document` être le résultat de [la création d'un nouveau contexte de navigation et d'un document](#) avec `null`, `null` et `group` .
4. [Ajoutez](#) le contexte de navigation au groupe .
5. Retourner le groupe et le document .

Pour **ajouter** un [contexte de navigation de niveau supérieur](#) `browserContext` à un groupe [de groupes de contextes de navigation](#) :

1. [Ajoute](#) `scanningContext` à [l'ensemble de contextes de navigation](#) du groupe .
2. Définissez le [groupe](#) de `scanningContext` sur `group` .

Pour **supprimer** un [contexte de navigation de niveau supérieur](#) `browserContext` :

1. [Assert](#) : le [groupe](#) de `scanningContext` n'est pas nul.
2. Soit `group` le [groupe](#) de `browserContext` .
3. Définissez le [groupe](#) de `scanningContext` sur `null`.
4. [Supprimez](#) `browserContext` de [l'ensemble de contextes de navigation](#) du groupe .
5. Si [l'ensemble de contextes de navigation](#) du groupe [est vide](#) , alors [supprimez](#) le groupe de [l'ensemble de groupes de contextes de navigation](#) de l'agent utilisateur .

*[Ajouter](#) et [supprimer](#) sont des opérations primitives qui aident à définir la durée de vie d'un [groupe de contexte de navigation](#) . Ils sont appelés par des opérations de création et de destruction de niveau supérieur pour [Document](#) les s et [les contextes de navigation](#) .*

Lorsqu'il n'y a aucun [Document](#) objet dont [le contexte de navigation est égal à un contexte de navigation](#) donné (c'est-à-dire que tous ces [Documents](#) ont été [détruits](#) ) et que le contexte [de navigation](#)`WindowProxy` est éligible pour la récupération de place, alors le [contexte de navigation](#) ne sera plus jamais accessible. S'il s'agit d'un [contexte de navigation de niveau supérieur](#) , à ce stade, l'agent utilisateur doit le [supprimer](#) .

### 7.3.3 Documents pleinement actifs

Un [Document](#) *d* est dit **entièrement actif** lorsque *d* est le [document actif](#) d'un [nœud navigable](#) *navigable*, et soit *navigable* est un [document traversable de niveau supérieur](#), soit le [document conteneur](#) de *navigable* est [entièrement actif](#).

Parce qu'ils sont associés à un élément, les [éléments navigables enfants](#) sont toujours liés à un document spécifique [Document](#), leur [document conteneur](#), dans leur [élément navigable parent](#). Les agents utilisateurs ne doivent pas permettre à l'utilisateur d'interagir avec des [éléments navigables enfants](#) dont les [documents conteneurs](#) ne sont pas eux-mêmes [pleinement actifs](#).

L'exemple suivant illustre comment a [Document](#) peut être le [document actif](#) de son [nœud navigable](#), tout en n'étant pas [entièrement actif](#). Here `a.html` est chargé dans une fenêtre de navigateur, `b-1.html` commence par être chargé dans un [iframe](#) comme indiqué, et `b-2.html` et `c.html` sont omis (il peut simplement s'agir d'un document vide).

```
<!-- a.html -->
<!DOCTYPE html>
<html lang="en">
<title>Navigable A</title>

<iframe src="b-1.html"></iframe>
<button onclick="frames[0].location.href = 'b-2.html'">Click
me</button>

<!-- b-1.html -->
<!DOCTYPE html>
<html lang="en">
<title>Navigable B</title>

<iframe src="c.html"></iframe>
```

A ce stade, les documents donnés par `a.html`, `b-1.html`, et `c.html` sont tous les [documents actifs](#) de leurs [nœuds navigables](#) respectifs. Ils sont également tous [pleinement actifs](#).

Après avoir cliqué sur le [button](#), et donc chargé un nouveau [Document](#) from `b-2.html` dans le navigable B, nous avons les résultats suivants :

- Le `a.html` [Document](#) reste à la fois le [document actif](#) de navigable A, et [pleinement actif](#).
- Le *n'est* `b-1.html` [Document](#) plus le [document actif](#) de navigable B. En tant que tel, il n'est pas non plus [entièrement actif](#).

- Le nouveau `b-2.html Document` est maintenant le [document actif](#) de navigable B, et est également [entièrement actif](#) .
- Le `c.html Document` est toujours le [document actif](#) du C navigable. Cependant, puisque [le document conteneur](#) de C est le `b-1.html Document`, qui n'est lui-même pas [complètement actif](#) , cela signifie que le `c.html Document` n'est plus [complètement actif](#) .

## 7.4 Historique de navigation et de session

Bienvenue dans la gueule du dragon. La navigation, l'historique de session et la traversée de cet historique de session sont quelques-unes des parties les plus complexes de cette norme.

Le concept de base peut ne pas sembler si difficile :

- L'utilisateur regarde un [navigable](#) qui présente son [document actif](#) . Ils [naviguent](#) vers une autre [URL](#) .
- Le navigateur récupère l'URL donnée sur le réseau, l'utilisant pour [remplir](#) une nouvelle [entrée d'historique de session](#) avec une nouvelle [entrée Document](#) .
- Le navigateur met à jour l' [entrée d'historique de session active](#) du [navigable](#) vers celle qui vient d'être remplie, et met ainsi à jour le [document actif](#) qu'il montre à l'utilisateur.
- À un moment donné plus tard, l'utilisateur [appuie sur le bouton de retour du navigateur](#) pour revenir à l' [entrée précédente de l'historique de la session](#) .
- Le navigateur consulte l' [URL](#) stockée dans cette [entrée d'historique de session](#) et l'utilise pour récupérer à nouveau et [remplir le document](#) de cette entrée .
- Le navigateur met à jour à nouveau l' [entrée d'historique de session active](#) de l' [élément navigable](#) .

Vous pouvez voir une partie de la complexité entrelacée ici, dans la façon dont la traversée peut provoquer une navigation (c'est-à-dire une récupération du réseau vers une URL stockée), et comment une navigation doit nécessairement s'interfacer avec la liste d'historique de session pour s'assurer que lorsqu'elle se termine l'utilisateur regarde la bonne chose. Mais les vrais problèmes surviennent avec les différents cas périphériques et les fonctionnalités de la plate-forme Web en interaction :

- [Les éléments navigables enfants](#) (par exemple, ceux contenus dans `iframes`) peuvent également naviguer et traverser, mais ces navigations doivent être linéarisées dans [une seule liste d'historique de session](#) puisque l'utilisateur n'a

qu'une seule interface avant/arrière pour l'ensemble de [l'élément navigable traversable](#) (par exemple, l'onglet du navigateur ).

- Étant donné que l'utilisateur peut parcourir plus d'une étape dans l'historique de la session (par exemple, en maintenant enfoncé son bouton de retour), il peut finir par parcourir plusieurs éléments [navigables](#) en même temps lorsque [des éléments navigables enfants](#) sont impliqués. Cela doit être synchronisé sur tous les navigables impliqués, ce qui peut impliquer plusieurs [boucles d'événements](#) ou même [des clusters d'agents](#) .
- Pendant la navigation, les serveurs peuvent répondre avec des codes d'état 204 ou 205 ou avec `Content-Disposition: attachment` des en-têtes `` , qui provoquent l'abandon de la navigation et le [navigable](#) reste sur son [document actif](#) d'origine . (C'est bien pire si cela se produit lors d'une navigation initiée par traversée !)
- Divers autres en-têtes HTTP, tels que ``Location``, ``Refresh``, ``X-Frame-Options`` et ceux de la politique de sécurité du contenu, contribuent soit au [processus de récupération](#) , soit au `Document`[processus de création](#) , soit aux deux. L' `Cross-Origin-Opener-Policy` en-tête `` contribue même au processus [de sélection et de création du contexte de navigation](#) !
- Certaines navigations (à savoir [les navigations fragmentées](#) et [les navigations d'application sur une seule page](#) ) sont synchrones, ce qui signifie que le code JavaScript s'attend à observer instantanément les résultats de la navigation. Cela doit ensuite être synchronisé avec la vue de l'historique de la session que voient tous les autres [éléments navigables](#) de l'arborescence, ce qui peut être soumis à des conditions de concurrence et nécessiter la résolution de vues conflictuelles de l'historique de la session.
- La plate-forme a accumulé diverses fonctionnalités passionnantes liées à la navigation qui nécessitent une casse spéciale, telles que `javascript:` les URL, `srcdoc` `iframe` les s et l' `beforeunload` événement.

Dans ce qui suit, nous avons tenté de guider le lecteur à travers ces complexités en les délimitant de manière appropriée dans des sections et des algorithmes étiquetés, et en donnant des mots d'introduction appropriés lorsque cela est possible. Néanmoins, si vous souhaitez bien comprendre la navigation et l'historique des sessions, [les conseils habituels](#) vous seront précieux.

## 7.4.1 Historique des sessions

### 7.4.1.1 Entrées de l'historique des sessions

Une **entrée d'historique de session** est une [structure](#) avec les [éléments](#) suivants :

- **step** , un entier non négatif ou " `pending` ", initialement " `pending` ".

- **URL** , une [URL](#)
- **état du document** , un [état du document](#) .
- **état sérialisé** , qui est [un état sérialisé](#) , initialement [StructuredSerializeForStorage](#) (null).
- **mode de restauration de défilement** , un [mode de restauration de défilement](#) , initialement " [auto](#)".
- **données de position de défilement** , qui sont des données de position de défilement pour les [régions de défilement restaurables](#) du [document](#) .
- **état utilisateur persistant** , qui est [défini par l'implémentation](#) , initialement null

Par exemple, certains agents utilisateurs peuvent vouloir conserver les valeurs des contrôles de formulaire.

*dir Les agents utilisateurs qui conservent la valeur des contrôles de formulaire sont encouragés à conserver également leur directionnalité (la valeur de l'attribut de l'élément ). Cela empêche les valeurs d'être affichées de manière incorrecte après un parcours d'historique lorsque l'utilisateur avait initialement saisi les valeurs avec une directionnalité explicite et non par défaut.*

Pour obtenir le **document** d' une [entrée d'historique de session](#) , retournez [le document](#) de son [état de document](#) .

---

**L'état sérialisé** est une sérialisation (via [StructuredSerializeForStorage](#) ) d'un objet représentant un état de l'interface utilisateur. Nous nous référons parfois de manière informelle aux "objets d'état", qui sont les objets représentant l'état de l'interface utilisateur fournis par l'auteur, ou alternativement les objets créés en désérialisant (via [StructuredDeserialize](#) ) l'état sérialisé.

Les pages peuvent [ajouter un état sérialisé](#) à l'historique de la session. Celles-ci sont ensuite [désérialisées](#) et [renvoyées au script](#) lorsque l'utilisateur (ou le script) remonte dans l'historique, permettant ainsi aux auteurs d'utiliser la métaphore "navigation" même dans les applications d'une page.

*L'état sérialisé est destiné à être utilisé à deux fins principales : premièrement, stocker une description préparée de l'état dans l' [URL](#) afin que, dans le cas simple, un auteur n'ait pas à effectuer l'analyse (bien que l'on aurait toujours besoin de l'analyse pour la gestion [URL](#) transmises par les utilisateurs, il ne s'agit donc que d'une optimisation mineure). Deuxièmement, pour que l'auteur puisse stocker un état*

que l'on ne stockerait pas dans l'URL car il ne s'applique qu'à l' Document instance actuelle et il devrait être reconstruit si un nouveau Document était ouvert.

Un exemple de ce dernier serait quelque chose comme garder une trace de la coordonnée précise à partir de laquelle un popup div a été créé pour s'animer, de sorte que si l'utilisateur revient en arrière, il peut être amené à s'animer au même endroit. Ou bien, il pourrait être utilisé pour conserver un pointeur dans un cache de données qui seraient extraites du serveur en fonction des informations contenues dans l' URL . de sorte que lors des allers-retours, les informations n'aient pas à être extraites à nouveau.

---

Un **mode de restauration de défilement** indique si l'agent utilisateur doit restaurer la position de défilement persistante (le cas échéant) lors de la traversée vers une entrée . Un mode de restauration de défilement est l'un des suivants :

" auto "

L'agent utilisateur est responsable de la restauration de la position de défilement lors de la navigation.

" manual "

La page est responsable de la restauration de la position de défilement et l'agent utilisateur n'essaie pas de le faire automatiquement

#### 7.4.1.2 État des documents

L'**état du document** contient l'état dans une entrée d'historique de session concernant la manière de présenter et, si nécessaire, de recréer un fichier Document. Il a:

- Un **document** , a Document ou null, initialement null.

Lorsqu'une entrée d'historique est active , elle a un Document dans son état de document . Cependant, lorsqu'un Document n'est pas complètement actif , il est possible qu'il soit détruit pour libérer des ressources. Dans de tels cas, cet élément de document sera annulé. L' URL et d'autres données dans l' entrée de l'historique de session et l'état du document sont ensuite utilisées pour créer une nouvelle Document à la place de l'original, dans le cas où l'agent utilisateur se trouve obligé de traverser jusqu'à l'entrée.

Si le Document n'est pas détruit , alors pendant le parcours de l'historique , il peut être réactivé . Le cache dans lequel les navigateurs stockent



ces `Documents` est souvent appelé un cache arrière ou bfcache (ou peut-être un cache « incroyablement rapide » ).

- Un **conteneur de politique d'historique** , un [conteneur de politique](#) ou null, initialement null.
- Un **réfèrent de requête** , qui est " `no-referrer`", " `client`", ou une [URL](#) , initialement " `client`".
- Une **stratégie de réfèrent de demande** , qui est une [stratégie de réfèrent](#) , initialement la [stratégie de réfèrent par défaut](#) .

La [stratégie de référence de demande](#) est distincte de la [stratégie de référence](#) du [conteneur de stratégie d'historique](#) . Le premier est utilisé pour les récupérations de ce document, tandis que le second contrôle les récupérations par ce document.

- Un **initiateur origin** , qui est une [origine](#) ou null, initialement null.
- Une **origin** , qui est une [origin](#) ou null, initialement null.

Il s'agit de l'origine sur laquelle nous avons défini [l'origine](#)<sub>about</sub>: de " `-schemed Documents` ". Nous le stockons ici car il est également utilisé lors de la restauration de ces s lors de la traversée, car ils sont reconstruits localement sans visiter le réseau. Il est également utilisé pour comparer l'origine avant et après le [remplissage de l'entrée d'historique de session](#) . Si les origines changent, le [nom de la cible navigable](#) est effacé.`Document`

- **Historiques imbriqués** , une [liste](#) d ' [historiques imbriqués](#) , initialement une [liste](#) vide .
- Une **ressource** , une chaîne, [une ressource POST](#) ou null, initialement null.

Une chaîne est traitée comme HTML. Il est utilisé pour stocker la source d'un `iframe_srcdoc`[document](#) .

- Un booléen **en attente de rechargement** , initialement faux.
- Un booléen **toujours rempli** , initialement faux.
- Une chaîne **de nom de cible navigable** , initialement la chaîne vide.

Les agents utilisateurs peuvent [détruire](#) les [documents](#) des [états de document avec des documents](#) non nuls , tant que le `Document` n'est pas [complètement actif](#) .

En dehors de cette restriction, cette norme ne spécifie pas quand les agents utilisateurs doivent détruire le [document](#) stocké dans un [document state](#) , plutôt que de le garder en cache.

---



Une **ressource POST** a :

- Un **corps de requête** , une [séquence d'octets](#) ou un échec.

Il n'y a accès qu'en [parallèle](#) , il n'est donc pas nécessaire de le stocker en mémoire. Cependant, il doit renvoyer la même [séquence d'octets](#) à chaque fois. Si cela n'est pas possible en raison de la modification des ressources sur le disque ou si les ressources ne sont plus accessibles, cela doit être défini sur échec.

- Un **type de contenu de requête** , qui est ``application/x-www-form-urlencoded``, ``multipart/form-data`` ou ``text/plain``.

---

Un **historique imbriqué** a :

- Un **identifiant** , une [valeur interne unique](#) .

*Ceci est utilisé pour associer l' [historique imbriqué](#) à un [navigable](#) .*

- **Entrées** , une [liste](#) d' [entrées d' historique de session](#) .

Cela contiendra plus tard des moyens d'identifier un enfant navigable à travers les rechargements.

---

Plusieurs entrées contiguës dans un historique de session peuvent partager le même [état de document](#) . Cela peut se produire lorsque l'entrée initiale est atteinte via [la navigation](#) normale et que l'entrée suivante est ajoutée via `history.pushState()` . Ou cela peut se produire via [la navigation vers un fragment](#) .

*Toutes les entrées qui partagent le même [état de document](#) (et qui sont donc simplement des états différents d'un document particulier) sont contiguës par construction.*

---

A [Document](#) a une **dernière entrée** , une [entrée d'historique de session](#) ou null.

*Il s'agit de l'entrée la plus récemment représentée par un [Document](#). Une seule [Document](#) peut représenter plusieurs [entrées d'historique de session](#) au fil du temps, car de nombreuses [entrées d'historique de session](#) contiguës peuvent partager le même [état de document](#) , comme expliqué ci-dessus.*

#### 7.4.1.3 Modifications centralisées de l'historique des sessions

Pour conserver une source unique de vérité, toutes les modifications apportées aux [entrées d'historique de session](#) d'un [élément navigable traversable](#) doivent être synchronisées. Ceci est particulièrement important en raison de la façon dont l'historique de session est influencé par tous les [navigables](#) descendants , et donc par de multiples [boucles d'événements](#) . Pour ce faire, nous utilisons la structure [de file d'attente parallèle de parcours d'historique de session](#) .

Une **file d'attente parallèle de parcours d'historique de session** est très similaire à une [file d'attente parallèle](#) . Il a un **ensemble d'algorithmes** , un [ensemble ordonné](#) .

Les [éléments](#) de [l'ensemble d'algorithmes](#) d' une [file d'attente parallèle de traversée d'historique de session](#) sont soit des étapes d'algorithme, soit **des étapes de navigation synchrones** , qui sont une marque particulière d'étapes d'algorithme impliquant une **cible navigable** (une [navigable](#) ).

Pour **ajouter des étapes de parcours de l'historique de session** à un algorithme d'étapes d' *étapes de l'algorithme* [traversable](#) [navigable](#) *traversable* donné , *ajoutez* des étapes à l'ensemble d' [algorithmes](#) de *la* [file d'attente de parcours de l'historique de session du](#) *traversable* .

Pour **ajouter des étapes de navigation synchrone de l'historique de session** impliquant une *cible* [navigable](#) *Navigable* à un algorithme [traversable](#) [navigable](#) *traversable* donné , *ajoutez* des étapes en tant *qu'étapes* [de navigation synchrone](#) ciblant *la cible navigable* *cible* *Navigable* à l' ensemble d' [algorithmes de la](#) [file d'attente de](#) *traversée* de l' [historique de session traversable](#) .

Pour **démarrer une nouvelle file d'attente parallèle de parcours d'historique de session** :

1. Soit *sessionHistoryTraversalQueue* une nouvelle [file d'attente parallèle de parcours d'historique de session](#) .
2. Exécutez les étapes suivantes [en parallèle](#) :

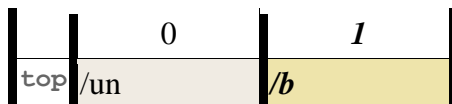
1. Alors que c'est vrai :

1. Si [l'ensemble d'algorithmes](#) de `sessionHistoryTraversalQueue` est vide, [continuez](#) .
2. Supposons que *les étapes* soient le résultat du [retrait](#) de [l'ensemble d'algorithmes](#) de `sessionHistoryTraversalQueue` .
3. Exécutez *les étapes* .

3. Renvoie `sessionHistoryTraversalQueue` .

[Les étapes de navigation synchrones](#) sont étiquetées dans l' [ensemble d'algorithmes](#) pour leur permettre de "sauter la file d'attente" conditionnellement. Ceci est géré dans [l'étape d'application de l'historique](#) .

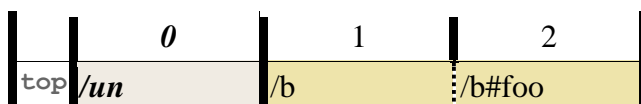
Imaginez l'historique des sessions conjointes représenté par ce [diagramme de Jake](#) :



Et le code suivant s'exécute au niveau supérieur :

```
history.back();  
location.href = '#foo';
```

Le résultat souhaité est :



Ce n'est pas simple, car la navigation synchronisée remporte la course en termes d'observabilité, tandis que la traversée remporte la course en termes d'étapes de mise en file d'attente sur la [file d'attente parallèle de traversée de l'historique de session](#) . Pour arriver à ce résultat, il se passe ce qui suit :

1. `history.back()` [ajoute des étapes](#) destinées à être parcourues par un delta de -1.
2. `location.href = '#foo'` modifie de manière synchrone l' entrée [d'entrée d'historique de session active](#) en une entrée nouvellement créée, avec l'URL `/b#foo`, et [ajoute des étapes synchrones](#) pour informer la source centrale de vérité de cette nouvelle entrée. Notez que cela ne met *pas* encore à jour l' [entrée de l'historique de la session en cours](#) , [l'étape de l'historique de la session en cours](#) ou la liste [des entrées de l'historique de la session](#) ; ces mises à jour ne peuvent pas être effectuées de manière synchrone et doivent être effectuées dans le cadre des étapes en file d'attente.

3. Dans la [file d'attente parallèle de parcours de l'historique de session](#) , les étapes mises en file d'attente par `history.back()` s'exécutent :

1. L'étape d'historique cible est déterminée comme étant 0 : l' [étape d'historique de la session en cours](#) (c'est-à-dire 1) plus le delta prévu de -1.
2. Nous entrons dans l' [application principale de l'](#) algorithme d'étape d'historique.

L'entrée à l'étape 0, pour l' `/a`URL, a son [document rempli](#) .

Pendant ce temps, la file d'attente est vérifiée pour [les étapes de navigation synchrones](#) . Les étapes mises en file d'attente par le `location.href` setter s'exécutent maintenant et empêchent la traversée d'effectuer des effets au-delà du remplissage de documents (tels que le déchargement de documents et la commutation d'entrées d'historique actives) jusqu'à ce qu'elles soient terminées. Ces étapes entraînent les événements suivants :

1. L'entrée avec l'URL `/b#foo` est ajoutée, avec son [étape](#) déterminée comme étant 2 : l' [étape de l'historique de la session en cours](#) (c'est-à-dire 1) plus 1.
2. Nous passons entièrement à cette entrée nouvellement ajoutée, y compris un appel imbriqué pour [appliquer l'étape d'historique](#) . Cela aboutit finalement à [la mise à jour du document](#) en envoyant des événements tels que `hashchange`.

Ce n'est qu'une fois que tout est terminé et que l' `/a`entrée d'historique a été entièrement remplie avec un [document](#) , que nous passons à l'application de l'étape d'historique étant donné l'étape cible de 0.

À ce stade, l' `Document`URL with `/b#foo` [se décharge](#) et nous terminons de passer à l'étape 0 de l'historique cible, ce qui fait que l'entrée with URL `/a` devient l' [entrée active de l'historique de la session](#) et 0 devient l' [étape actuelle de l'historique de la session](#) .

Voici un autre exemple plus complexe, impliquant des courses entre le remplissage de deux `iframe`s différents, et une navigation synchrone une fois l'un de ces iframes chargé. On commence avec cette configuration :

	0	1	2
top	/t		
frames[0]	/i-0-a	/i-0-b	
frames[1]	/i-1-a		/i-1-b

puis appelez `history.go(-2)` . Il se produit alors :

1. `history.go(-2)` [ajoute des étapes](#) destinées à traverser par un delta de -
2. Une fois ces étapes exécutées :

1. Le pas cible est déterminé comme étant  $2 + (-2) = 0$ .
2. En parallèle, les récupérations sont faites pour [remplir](#) les deux iframes, récupération `/i-0-a` et `/i-1-a` respectivement.

Pendant ce temps, la file d'attente est vérifiée pour [les étapes de navigation synchrones](#) . Il n'y en a pas pour le moment.

3. Dans la course au fetch, le fetch for `/i-0-a` gagne. Nous poursuivons pour terminer tout le travail [d'application de l'étape d'historique](#) sur l'impact de la traversée sur le `frames[0]` [navigable](#) , y compris la mise à jour de son [entrée d'historique de session active](#) vers l'entrée avec l'URL `/i-0-a`.
4. Avant la fin de la `/i-1-a` récupération, nous atteignons le point où [les scripts peuvent s'exécuter pour le document nouvellement créé](#) dans le [document actif](#) du `frames[0]` [navigable](#) . Certains de ces scripts s'exécutent :

```
location.href = '#foo'
```

Cela modifie de manière synchrone l'entrée [d'entrée d'historique de session active](#) `frames[0]` de la navigation en une entrée nouvellement créée, avec l'URL , et [ajoute des étapes synchrones](#) pour informer la source centrale de vérité de cette nouvelle entrée. `/i-0-a#foo`

Contrairement à l' [exemple précédent](#) , ces étapes synchrones ne "sautent pas la file d'attente" et ne mettent pas à jour le [traversable](#) avant que nous ayons terminé la récupération pour `/i-1-a`. C'est parce que le navigable en question, `frames[0]`, a déjà été modifié dans le cadre de la traversée, nous savons donc qu'avec l' [étape 2 de l'historique de la session actuelle](#) , l'ajout de la nouvelle entrée en tant qu'étape 3 n'a pas de sens.

5. Une fois la récupération `/i-1-a` terminée, nous procédons à la mise à jour de la `frames[1]` [navigation](#) pour la traversée, y compris la mise à jour de son [entrée d'historique de session active](#) vers l'entrée avec URL `/i-1-a`.
  6. Maintenant que les deux navigables ont fini de traiter le parcours, nous mettons à jour l' [étape d'historique de la session en cours](#) à l'étape cible de 0.
2. Nous pouvons maintenant traiter les étapes qui ont été mises en file d'attente pour la navigation synchrone :

1. L' `/i-0-a#foo` entrée est ajoutée, avec son étape déterminée comme étant 1 : l' étape de l'historique de la session en cours (c'est-à-dire 0) plus 1. Cela efface également l'historique de transfert existant .
2. Nous passons complètement à cette entrée nouvellement ajoutée, y compris en appelant apply the history step . Cela aboutit finalement à la mise à jour du document en envoyant des événements tels que `hashchange`, ainsi qu'à la mise à jour de l' étape actuelle de l'historique de la session à l'étape cible de 1.

Le résultat final est :

	0	1
top	/t	
frames[0]	/i-0-a	/i-0-a#foo
frames[1]	/i-1-a	

#### 7.4.1.4 Opérations de bas niveau sur l'historique des sessions

Cette section contient un ensemble d'opérations diverses que nous effectuons tout au long de la norme lors de la manipulation de l'historique des sessions. La meilleure façon d'avoir une idée de ce qu'ils font est de regarder leurs sites d'appels.

Pour **obtenir des entrées d'historique de session** pour un navigable , *navigable* :

1. Soit *traversable* le *navigable* traversable de navigable .
2. Assert : cela s'exécute dans la file d'attente de traversée de l'historique de session *de traversable* .
3. Si *navigable* est *traversable* , renvoie les entrées d' historique de session *de traversable* .
4. Soit *docStates* un ensemble ordonné vide d' états de document .
5. Pour chaque *entrée* des entrées d'historique de session *de traversable* , ajoutez l'état du document de *l'entrée* à *docStates* .
6. Pour chaque *docState* de *docStates* :
  1. Pour chaque *nestedHistory* des historiques imbriqués de *docState* :
    1. Si l' id de *nestedHistory* est égal à l' id de *navigable* , renvoie les entrées de *nestedHistory* .
    2. Pour chaque *entrée* des entrées de *nestedHistory* , ajoutez l'état du document de *l'entrée* à *docStates* .

7. [Assert](#) : cette étape n'est pas atteinte.

Pour **effacer l'historique de session** aller d'un [navigable traversable](#) *navigable* :

1. [Assert](#) : cela s'exécute dans la file d'attente de traversée de l'historique des [sessions](#) de *navigable* .
2. Soit *step* l' étape de l'historique de [la session en cours du](#) *navigable* .
3. Soit *entryLists* l'ensemble [ordonné](#) « [entrées de l'historique](#) de session de *navigable* ».
4. [Pour chaque](#) liste d'entrées de listes d'entrées :
  1. [Supprimez](#) chaque [entrée d'historique de session](#) de *entryList* dont le [pas](#) est supérieur à *step* .
  2. [Pour chaque](#) entrée de *entryList* :
    1. [Pour chaque](#) *nestedHistory* des [historiques imbriqués](#) de [l'état du document](#) d' une entrée , [ajoutez la liste d'entrées](#) de *nestedHistory* à *entryLists* .

Pour **obtenir toutes les étapes d'historique utilisées** qui font partie de [traversable](#) *traversable* :

1. [Assert](#) : cela s'exécute dans [la file d'attente de traversée](#) de l'historique de session de *traversable* .
2. Soit *étapes* un [ensemble ordonné vide](#) d'entiers non négatifs.
3. Soit *entryLists* l'ensemble [ordonné](#) « [entrées](#) de l'historique de session de *traversable* ».
4. [Pour chaque](#) liste d'entrées de listes d'entrées :
  1. [Pour chaque](#) entrée de *entryList* :
    1. [Ajouter l'étape](#) de l'entrée aux *étapes* .
    2. [Pour chaque](#) *nestedHistory* des [historiques imbriqués](#) de [l'état du document](#) d' une entrée , [ajoutez la liste d'entrées](#) de *nestedHistory* à *entryLists* .
5. *Étapes* de retour , [triées](#) .

Pour **appliquer les modifications d'historique en attente** à un [traversable](#) *traversable* avec un booléen facultatif *checkForUserCancelation* (false par défaut) :

1. Soit *targetStep* l' [étape courante de l'historique](#) de la session *traversable* .



2. [Appliquez l'étape d'historique](#) `targetStep` à traversable avec [checkForUserCancelation](#) défini sur `checkForUserCancelation`.

## 7.4.2 La navigation

Certaines actions amènent un [navigable](#) à [naviguer](#) vers une nouvelle ressource.

Par exemple, [suivre un lien hypertexte](#), [la soumission d'un formulaire](#) et les méthodes `window.open()` et `location.assign()` peuvent tous entraîner une navigation.

*Bien que dans cette norme, le mot "navigation" se réfère spécifiquement à l'algorithme [de navigation](#), cela ne correspond pas toujours aux perceptions des développeurs Web ou des utilisateurs. Par exemple:*

- *Les [étapes de mise à jour de l'URL et de l'historique](#) sont souvent utilisées lors des soi-disant "navigations d'application sur une seule page" ou "navigations dans le même document", mais elles ne déclenchent pas l'algorithme [de navigation](#).*
- *Les [rechargements](#) et les [parcours](#) sont parfois considérés comme un type de navigation, car tous les trois tenteront souvent [de remplir le document de l'entrée d'historique](#) et pourraient donc effectuer des récupérations de navigation. Voir, par exemple, les API exposées Navigation Timing. Mais ils ont leurs propres algorithmes de point d'entrée, distincts de l'algorithme [de navigation](#). [\[TEMPS DE NAVIGATION\]](#)*
- *Bien que les [navigations fragmentées](#) soient toujours effectuées via l'algorithme [de navigation](#), un utilisateur peut les percevoir davantage comme un saut d'une page à l'autre que comme une véritable navigation.*

### 7.4.2.1 Notions complémentaires

Avant de pouvoir sauter dans l'[algorithme de navigation](#) lui-même, nous devons établir plusieurs structures importantes qu'il utilise.

La **structure des paramètres de l'instantané source** est utilisée pour capturer les données d'un lancement de navigation. Il est instantané au début d'une navigation et utilisé tout au long de la durée de vie de la navigation. Il comporte les [éléments](#) suivants :`Document`

**a une activation transitoire**  
un booléen



### **drapeaux de bac à sable**

un [ensemble de drapeaux de sandboxing](#)

### **permet le téléchargement**

un booléen

### **recupérer le client**

un [objet de paramètres d'environnement](#) , à utiliser uniquement en tant que [client de requête](#)

### **conteneur de stratégie source**

un [conteneur de stratégie](#)

Pour **capturer les paramètres d'instantané source** donnés par un [Document](#) *sourceDocument* , renvoyez un nouveau [paramètre d'instantané source](#) avec

#### **[a une activation transitoire](#)**

true si [l'objet global pertinent](#) de *sourceDocument* a [une activation transitoire](#) ; sinon faux

#### **[drapeaux de bac à sable](#)**

[Ensemble d'indicateurs de sandboxing actif](#) de *sourceDocument*

#### **[permet le téléchargement](#)**

false si [l'indicateur de sandboxing actif](#) de *sourceDocument* a l' [indicateur de contexte de navigation des téléchargements en sandbox](#) défini ; sinon vrai

#### **[recupérer le client](#)**

[Objet de paramètres pertinent](#) de *sourceDocument*

#### **[conteneur de stratégie source](#)**

[Conteneur de stratégie](#) de *sourceDocument*

---

La [structure de paramètres d'instantané cible](#) est utilisée pour capturer les données d'un [élément navigable](#) en cours de navigation. Comme [les paramètres d'instantané source](#) , il est pris en instantané au début d'une navigation et utilisé tout au long de la durée de vie de la navigation. Il comporte les [éléments](#) suivants :

### **drapeaux de bac à sable**

un [ensemble de drapeaux de sandboxing](#)

Pour **créer un instantané des paramètres d'instantané cible** en fonction d'un *targetNavigable* [navigable](#) , renvoyez un nouveau [paramètre d'instantané cible](#) avec [des indicateurs de sandboxing](#) définis sur le résultat de [la détermination des indicateurs de sandboxing de création](#) en fonction [du contexte de navigation actif](#) de *targetNavigable* et du [conteneur de](#) *targetNavigable* .

---

Une grande partie du processus de navigation consiste à déterminer comment créer un nouveau [Document](#), ce qui se produit finalement lors de la [création et de l'initialisation d'un Document](#) algorithmne [d'objet](#). Les paramètres de cet algorithme sont suivis via une **navigation params** [struct](#), qui contient les [éléments](#) suivants :

**identifiant**

null ou un [ID de navigation](#)

**demande**

null ou une [requête](#) qui a lancé la navigation

**réponse**

une [réponse](#) qui a finalement été naviguée (potentiellement une [erreur réseau](#))

**origine**

une [origine](#) à utiliser pour le nouveau [Document](#)

**conteneur de stratégie**

un [conteneur de stratégie](#) à utiliser pour le nouveau [Document](#)

**ensemble final de drapeaux de sandboxing**

un [drapeau de sandboxing](#) à imposer au nouveau [Document](#)

**politique d'ouverture d'origine croisée**

une [stratégie d'ouverture d'origine croisée](#) à utiliser pour le nouveau [Document](#)

**Résultat de l'application COOP**

un [résultat d'application de la politique d'ouverture d'origine croisée](#), utilisé pour les rapports et potentiellement pour provoquer un [changement de groupe de contexte de navigation](#)

**environnement réservé**

null ou un [environnement](#) réservé au nouveau [Document](#)

**navigable**

le [navigable](#) à naviguer

**type de temps de navigation**

a [NavigationTimingType](#) utilisé pour [créer l'entrée de temps de navigation](#) pour le nouveau [Document](#)

**recupérer le contrôleur**

null ou un [contrôleur de récupération](#)

**valider les premiers indices**

null ou un algorithme acceptant a [Document](#), une fois créé

Une fois qu'une structure [de paramètres de navigation](#) est créée, cette norme ne modifie aucun de ses [éléments](#) . Ils ne sont transmis qu'à d'autres algorithmes.

---

Un **ID de navigation** est une chaîne UUID générée lors de la navigation. Il est utilisé pour s'interfacer avec la spécification *WebDriver BiDi* ainsi que pour suivre la [navigation en cours](#) . [\[PILOTE WEBBIDI\]](#)

---

Après [Document](#) la création, l' [historique de session](#) du [navigable traversable](#) pertinent est mis à jour. Un **comportement de gestion d'historique** est utilisé pour suivre le type souhaité de mise à jour d'historique de session tout au long du processus de navigation. Il s'agit de l'un des éléments suivants :

" **push** "

Une navigation régulière qui ajoute une nouvelle [entrée d'historique de session](#) et [effacera l'historique de session vers l'avant](#) .

" **replace** "

Une navigation qui remplacera l' [entrée active de l'historique de la session](#) .

#### 7.4.2.2 Commencer la navigation

Chaque [élément navigable](#) a une **navigation en cours** , qui est un [ID de navigation](#) , " `traversal` ", ou null, initialement null. Il est utilisé pour suivre l'abandon de la navigation et pour empêcher toute navigation d'avoir lieu pendant [la traversée](#) .

Pour **naviguer** dans une *URL* [navigable](#) *navigable* vers une [URL](#) à l'aide d'un *sourceDocument* , avec une [ressource POST](#) facultative , une chaîne ou **un documentResource** [null](#) (valeur nulle par défaut), une [réponse](#) optionnelle -ou- [null](#) (valeur nulle par défaut), une option booléenne **exceptionsEnabled** ( valeur par défaut false) , un [comportement de gestion de l'historique](#) [facultatif](#) **historyHandling** (par défaut " "), une chaîne facultative **cspNavigationType** (par défaut " ") et une [stratégie de référent](#) [facultative](#) **referrerPolicy** [Document](#) [push](#) [other](#) (par défaut la chaîne vide):

1. Soit *sourceSnapshotParams* le résultat de [l'instantané des paramètres de l'instantané](#) source donné *sourceDocument* .

2. Soit *initiatorOriginSnapshot* l' origine de *sourceDocument* .
3. Si le nœud navigable de *sourceDocument* n'est pas autorisé par le sandboxing à parcourir *navigable* given et *sourceSnapshotParams* , alors :
  1. Si *exceptionsEnabled* est vrai, lancez un `"_SecurityError"` `DOMException` .
  2. Retour.
4. Soit *navigationId* le résultat de la génération d'un UUID aléatoire . `[WEBCRYPTO]`
5. Si l' agent environnant est égal à l'agent pertinent du document actif de *navigable* , continuez ces étapes. Sinon, placez une tâche globale en file d'attente sur la source de la tâche de navigation et de traversée en fonction de la fenêtre active de *navigable* pour poursuivre ces étapes.

*Nous faisons cela parce que nous sommes sur le point d'examiner un grand nombre de propriétés du document actif de *navigable* , qui ne sont en théorie accessibles que dans la boucle d'événements appropriée . (Mais, nous ne voulons pas mettre une tâche en file d'attente de manière inconditionnelle, car, par exemple, les navigations de fragments de boucle d'événement identique doivent prendre effet de manière synchrone.)*

*Une autre stratégie de mise en œuvre consisterait à répliquer les informations pertinentes dans des boucles d'événements ou dans un "processus de navigateur" canonique, afin qu'elles puissent être consultées sans mettre une tâche en file d'attente. Cela pourrait donner des résultats différents de ceux que nous spécifions ici dans les cas extrêmes, où les propriétés pertinentes ont changé dans la boucle d'événements cible mais n'ont pas encore été répliquées. Des tests supplémentaires sont nécessaires pour déterminer laquelle de ces stratégies correspond le mieux au comportement du navigateur, dans de tels cas extrêmes.*

6. Si le compteur de déchargement du document actif de *navigable* est supérieur à 0, invoquez l'échec de la navigation WebDriver BiDi avec un statut de navigation WebDriver BiDi dont l'*id* est *navigationId* , le statut est " " et l'*url* est *url* , et retournez `.canceled`
7. Si l'une des conditions suivantes est vraie :
  1. *url* est égal à l'URL du document actif de *navigable* ;
  2. le schéma de l' *url* est `"javascript"` ; ou
  3. le document actif de *navigable* est initial`about:blank` est vraipuis définissez *historyHandling* sur `"replace"`.
8. Si toutes les conditions suivantes sont vraies :
  1. *documentResource* est nul ;
  2. *la réponse* est nulle ;

3. *url* est égal à l'URL de l'entrée d'historique de session active de navigable avec les fragments d'exclusion définis sur *true* ; et
4. le fragment de l' *url* n'est pas nul

alors:

5. Accédez à un fragment donné *navigable* , *url* , *historyHandling* et *navigationId* .
6. Retour.
9. Si le parent de *navigable* n'est pas nul, alors définissez *navigable* retarde les événements sur *true.load*
10. Soit *targetBrowsingContext* le contexte de navigation actif de *navigable* .
11. Soit *targetSnapshotParams* le résultat de l'instantané des paramètres d'instantané cible donnés *navigable* .
12. Appelez la navigation WebDriver BiDi démarrée avec *targetBrowsingContext* , et un nouveau statut de navigation WebDriver BiDi dont l'id est *navigationId* , l'url est *url* et le statut est "*pending*".
13. Si la navigation en cours de *navigable* est " " , alors :*traversal*
  1. L'appel de la navigation WebDriver BiDi a échoué avec *targetBrowsingContext* et un nouveau statut de navigation WebDriver BiDi dont l'id est *navigationId* , le statut est "*canceled*" et l'url est *url* .
  2. Retour.

*Toute tentative de navigation sur un navigable en cours de traversée est ignorée.*

14. Définissez la navigation continue de *navigable* sur *navigationId* .

*Cela aura pour effet d'interrompre d'autres navigations en cours de navigable , car à certains moments de la navigation, des modifications de la navigation en cours entraîneront l'abandon de travaux supplémentaires.*

15. Si le schéma de l' *url* est " " , alors :*javascript*
  1. Mettre en file d'attente une tâche globale sur la source de tâche de navigation et de traversée donnée de la fenêtre active de *navigable* pour naviguer vers une URL donnée *navigable* , *url* , *historyHandling* , *initiatorOriginSnaps* *hot* et *cspNavigationType* .*javascript*:
  2. Retour.
16. En parallèle , exécutez ces étapes :

1. Soit `unloadPromptCanceled` le résultat de [la vérification si le déchargement est annulé par l'utilisateur](#) pour les [navigables descendants inclusifs](#) du [document actif](#) de *navigable* .
2. Si `unloadPromptCanceled` est vrai, ou si [la navigation en cours](#) de *navigable* n'est plus `navigationId` , alors :
  1. L'appel [de la navigation WebDriver BiDi a échoué](#) avec `targetBrowsingContext` et un nouveau [statut de navigation WebDriver BiDi](#) dont `l'id` est `navigationId` , [le statut](#) est "`canceled`" et `l'url` est `url` .
  2. Abandonnez ces étapes.
3. [Mettre en file d'attente une tâche globale](#) sur la [source de tâches de navigation et de traversée](#) donnée [dans la fenêtre active](#) de *navigable* pour [abandonner le document actif](#) de *navigable* .
4. Soit `documentState` un nouvel [état de document](#) avec

**[demander la politique de référence](#)**

`referrerPolicy`

**[origine de l'initiateur](#)**

`initiatorOriginSnapshotinitiatorOriginSnapshot`

**[Ressource](#)**

`documentResource`

**[nom de la cible navigable](#)**

[nom cible](#) de *navigable*

*Le [nom de la cible navigable](#) peut être effacé dans diverses conditions plus tard dans le processus de navigation, avant que l'état du document ne soit finalisé.*

5. Si `url` est `about:blank`, alors définissez [l'origine](#) de `documentState` sur [l'origine](#) de l'initiateur de `documentState` .
6. Sinon, si `url` est `about:srcdoc`, alors définissez [l'origine](#) de `documentState` sur [l'origine](#) du document [actif](#) du [parent](#) de *navigable* .
7. Soit `historyEntry` une nouvelle [entrée d'historique de session](#) , avec son [URL](#) définie sur `url` et son [état de document](#) défini sur `documentState` .
8. Laissez `navigationParams` être nul.
9. Si *la réponse* est non nulle :

*L' [algorithme de navigation](#) n'est fourni avec une [réponse](#) que dans le cadre des modèles de traitement `object` et `embed`, ou pour traiter des parties de `multipart/x-mixed-replace` [réponses](#) après la réponse initiale.*

1. Soit *policyContainer* le résultat de [la détermination du conteneur de stratégie des paramètres de navigation](#) en fonction de l' [URL de la réponse](#) , null , un [clone](#) du [conteneur de stratégie](#) de *sourceDocument* , du [conteneur de stratégie](#) du [document](#) de conteneur *navigable* et de la valeur null .
2. Soit *finalSandboxFlags* l' [union](#) des [indicateurs de sandboxing](#) de *targetSnapshotParams* et [des indicateurs de sandboxing dérivés](#) du [CSP](#) de la liste CSP de *policyContainer* .
3. Soit *responseOrigin* le résultat de [la détermination de l'origine](#) en fonction de [l'URL](#) de la réponse , des *finalSandboxFlags* , de l'origine de [l'initiateur de](#) *documentState* et de null.
4. Laissez *coop* être une nouvelle [politique d'ouverture d'origine croisée](#) .
5. Soit *coopEnforcementResult* un nouveau [résultat d'application de la politique d'ouverture d'origine croisée](#) avec

[URL](#)  
[URL](#) de la réponse  
[origine](#)  
*réponseOrigine*  
[politique d'ouverture d'origine croisée](#)  
*coopérative*

6. Définissez *navigationParams* sur un nouveau [paramètre de navigation](#) , avec

[identifiant](#)  
*ID de navigation*  
[demande](#)  
 nul  
[réponse](#)  
*réponse*  
[origine](#)  
*réponseOrigine*  
[conteneur de stratégie](#)  
*policyContainer*  
[ensemble final de drapeaux de sandboxing](#)  
*finalSandboxFlagsfinalSandboxFlags*  
[politique d'ouverture d'origine croisée](#)  
*coopérative*  
[Résultat de l'application COOP](#)  
*coopEnforcementResult*  
[environnement réservé](#)  
 nul  
[navigable](#)  
*navigable*



"[navigate](#)" [type de temps de navigation](#)

nul [récupérer le contrôleur](#)

nul [valider les premiers indices](#)

10. [Essayez de remplir le document de l'entrée d'historique](#) pour *historyEntry*, étant donné *navigable*, "[navigate](#)", *sourceSnapshotParams*, *targetSnapshotParams*, *navigationalId*, *navigationParams*, *cspNavigationType*, avec [allowPOST](#) défini sur true et [CompletionSteps](#) défini sur l'étape suivante :

1. [Ajoutez les étapes de parcours de l'historique de session](#) à [traversable](#) de *navigable* pour [finaliser une navigation inter-documents](#) en fonction de *navigable*, *historyHandling* et *historyEntry*.

### 7.4.2.3 Terminer la navigation

Bien que le cas habituel de navigation entre documents consiste d'abord à [remplir une entrée d'historique de session](#) avec un [Document](#), toutes les navigations qui ne sont pas abandonnées finiront par appeler l'un des algorithmes ci-dessous.

#### 7.4.2.3.1 Le cas habituel de navigation entre documents

Pour **finaliser une navigation inter-documents** en fonction d'un [navigable](#) *navigable*, [d'un comportement de gestion de l'historique](#) *historyHandling* et [d'une entrée d'historique de session](#) *historyEntry* :

1. [Assert](#) : cela s'exécute sur [la file d'attente de traversée de l'historique de session](#) de [navigable](#) *navigable*.
2. Définir *navigable* [retardeLoad](#) les [événements](#) sur false.
3. Si [le document](#) de *historyEntry* est nul, alors retournez.

*Cela signifie que [tenter de remplir le document de l'entrée d'historique](#) a fini par ne pas créer de document, par exemple, la navigation étant annulée par une navigation ultérieure, une réponse 204 No Content, etc.*

4. Si toutes les conditions suivantes sont vraies :
  - [le parent](#) de *navigable* est null ;



- [Le contexte de navigation](#) du [document](#) de *historyEntry* n'est pas un [contexte de navigation auxiliaire](#) dont [le contexte de navigation d'ouverture](#) n'est pas nul ; et
- [L'origine](#) du document de [historyEntry](#) n'est pas *navigable* [L'origine](#) du [document actif](#) de l'entrée

puis définissez le nom de [la cible navigable de l'état du document](#) de *historyEntry* sur la chaîne vide.

5. Soit *entryToReplace* l'entrée d' [historique de session active](#) de *navigable* si *historyHandling* est " ", sinon null. [replace](#)
6. Soit *traversable* le *navigable* traversable [de navigable](#) .
7. Laissez *targetStep* être nul.
8. Laissez *targetEntries* être le résultat de [l'obtention des entrées d'historique de session](#) pour *navigable* .
9. Si *entryToReplace* est nul, alors :
  - [Effacez l'historique de session avant](#) de *traversable* .
  - Définissez *targetStep* sur l'étape d'historique de la [session actuelle](#) de *traversable* + 1.
  - Définissez [le pas](#) de *historyEntry* sur *targetStep* .
  - [Ajoutez](#) *historyEntry* à *targetEntries* .

Sinon:

- [Remplacez](#) *entryToReplace* par *historyEntry* dans *targetEntries* .
  - Définissez le [pas](#) de *historyEntry* sur le [pas](#) de *entryToReplace* .
  - Définissez *targetStep* sur [l'étape d'historique de la session en cours](#) de *traversable* .
10. [Appliquez l'étape d'historique](#) *targetStep* à *traversable* .

#### 7.4.2.3.2 Le **javascript:** cas particulier de l'URL

**javascript:** Les URL ont une [étiquette dédiée](#) sur le suivi des problèmes documentant divers problèmes avec leur spécification.

Pour **naviguer vers une `javascript:URL`** , étant donné une cible navigable`Navigable` , une URL `url` , un comportement de gestion de l'historique `historyHandling` , un initiateur d'origine`Origin` et une chaîne `cspNavigationType` :

1. Assert : `historyHandling` est "`replace`".
2. Définissez la navigation continue de `targetNavigable` sur `null`.
3. Si `initiatorOrigin` n'est pas le même domaine d'origine que l'origine du document actif de `targetNavigable` , alors retournez.
4. Soit `request` une nouvelle requête dont l'URL est `url` .

*Il s'agit d'une demande synthétique uniquement pour la plomberie dans l'étape suivante. Il n'atteindra jamais le réseau.*

5. Si le résultat de la demande de navigation de type doit être bloqué par la politique de sécurité du contenu ? `requête` donnée et `cspNavigationType` est "`Blocked`", puis retour. [CSP]
6. Soit `newDocument` le résultat de l'évaluation d'une `javascript:URL` donnée `targetNavigable` , `url` et `initiatorOrigin` .
7. Si `newDocument` est `null`, alors retournez.

*Dans ce cas, du code JavaScript a été exécuté, mais aucun nouveau n'a `Document` été créé, nous n'effectuerons donc pas de navigation.*

8. Soit `entryToReplace` l' entrée active de l'historique de session de `targetNavigable` .
9. Soit `oldDocState` l' état du document `entryToReplace` .
10. Soit `documentState` un nouvel état de document avec

document

*nouveau document*

conteneur de stratégie d'historique

un clone du conteneur de politique d'historique de `oldDocState` s'il n'est pas nul ; nul sinon

réfèrent de la demande

réfèrent de requête de `oldDocState`

demande la politique de référence

La politique de référence de la demande de `oldDocState` **ou devrait-elle être**

**la politique de référence qui a été transmise pour naviguer ?**

origine

`initiatorOriginSnapshot``initiatorOriginSnapshot`

Ressource

nul

toujours peuplé

vrai

[nom de la cible navigable](#)

[nom de la cible navigable](#) de *oldDocState*

11. Soit *historyEntry* une nouvelle [entrée d'historique de session](#) , avec

[URL](#)

[URL](#) de *entryToReplace*

[état des documents](#)

*documentState*

Pour l' [URL](#) , nous n'utilisons pas *url* , *javascript:* c'est-à-dire l' URL réelle avec laquelle l'algorithme [de navigation](#) a été appelé. Cela signifie que *javascript:* les URL ne sont jamais stockées dans l'historique de session et ne peuvent donc jamais être parcourues.

12. [Ajoutez les étapes de parcours de l'historique de session](#) au [traversable](#) de *targetNavigable* pour [finaliser une navigation entre documents](#) avec *targetNavigable* , *historyHandling* et *historyEntry* .

Pour **évaluer une *javascript:URL*** en fonction d'une cible [navigableNavigable](#) , d'une [URL](#) *url* et d'une [origine](#) *newDocumentOrigin* :

1. Soit *urlString* le résultat de l'exécution du [sérialiseur d'URL](#) sur *url* .
2. Soit *encodedScriptSource* le résultat de la suppression du " *javascript:*" de tête de *urlString* .
3. Soit *scriptSource* le [décodage UTF-8](#) du [pourcentage de décodage](#) de *encodedScriptSource* .
4. Soit *settings* l' [objet de paramètres pertinent](#) du [document actif](#) de *targetNavigable* .
5. Soit *baseURL* l' URL de base de l' [API](#) des paramètres .
6. Soit *script* le résultat de [la création d'un script classique](#) avec *scriptSource* , *settings* , *baseURL* et les [options de récupération de script classique par défaut](#) .
7. Soit *evaluationStatus* le résultat de l'exécution du script [script classique](#) .
8. Soit *résultat* nul.
9. Si *evaluationStatus* est un achèvement normal et *evaluationStatus* .[[Value]] est une chaîne, alors définissez *result* sur *evaluationStatus* .[[Value]].
10. Sinon, renvoie null.
11. Soit *réponse* une nouvelle [réponse](#) avec

[URL](#)

[URL](#) du [document actif](#) de *targetNavigable*

### liste d'en-tête

« (Content-Type`, `text/html; charset=utf-8`) »

### corps

l'encodage UTF-8 de result , en tant que corps

L'encodage en UTF-8 signifie que les substitués non appariés n'effectueront pas d'aller-retour une fois que l'analyseur HTML aura décodé le corps de la réponse.

12. Soit *policyContainer* le conteneur de stratégie du document actif de *targetNavigable* .
13. Soit *finalSandboxFlags* les indicateurs de sandboxing dérivés de la liste CSP de *policyContainer* .
14. Soit *coop* la politique d' ouverture cross - origin du document actif de *targetNavigable* .
15. Soit *coopEnforcementResult* un nouveau résultat d'application de la politique d'ouverture d'origine croisée avec

### URL

URL

### origine

*newDocumentOrigin*

### politique d'ouverture d'origine croisée

*coopérative*

16. Soit *navigationParams* un nouveau navigation params , avec

### identifiant

*ID de navigation*

### demande

null cela entraînera la nullité du référent du résultat ; *Document* Est-ce exact?

### réponse

*réponse*

### origine

*newDocumentOrigin*

### conteneur de stratégie

*policyContainer*

### ensemble final de drapeaux de sandboxing

*finalSandboxFlags**finalSandboxFlags*

### politique d'ouverture d'origine croisée

*coopérative*

### Résultat de l'application COOP

*coopEnforcementResult*

### environnement réservé

nul

### navigable

*cibleNavigable*

### type de temps de navigation

"*navigate*"

[récupérer le contrôleur](#)

nul

[valider les premiers indices](#)

nul

17. Renvoie le résultat du [chargement d'un document HTML](#) donné *navigationParams* .

#### 7.4.2.3.3 Navigations fragmentaires

Pour **naviguer vers un fragment** donné un [navigable navigable](#) , une [URL url](#) , un [comportement de gestion de l'historique](#) *historyHandling* et un [ID de navigation](#) *navigationId* :

1. Soit *historyEntry* une nouvelle [entrée d'historique de session](#) , avec  
[URL](#)  
*URL*  
[état des documents](#)  
*état du document* de l' [entrée active de l'historique de session](#) de *navigable*  
[défilement mode de restauration](#)  
*mode de restauration* du défilement de [l'entrée active de l'historique de session](#) de *navigable*
2. Soit *entryToReplace* l'entrée d' [historique de session](#)  
[active](#) de *navigable* si *historyHandling* est " " , sinon null. *replace*
3. Soit *history* l' [objet historique](#) du [document actif](#) de *navigable* .
4. Soit *scriptHistoryIndex* l' [index](#) de l'historique .
5. Soit *scriptHistoryLength* la [longueur](#) de l'historique .
6. Si *historyHandling* vaut " *push* " , alors :
  1. Définissez [l'état](#) de l' *historique* sur null.
  2. Incrémenter *scriptHistoryIndex* .
  3. Définissez *scriptHistoryLength* sur *scriptHistoryIndex* + 1.
7. Définissez [l'entrée d'historique de session](#)  
[active](#) de *navigable* sur *historyEntry* .
8. [Mettre à jour le document pour l'application d'étape d'historique](#) en fonction du [document actif](#) de *navigable* , *historyEntry* , *true* , *scriptHistoryIndex* et *scriptHistoryLength* .

Cet algorithme sera appelé deux fois à la suite d'une navigation sur un seul fragment : une fois de manière synchrone, où les valeurs de meilleure estimation `scriptHistoryIndex` et `scriptHistoryLength` sont définies, `history.state` sont annulées et divers événements sont déclenchés ; et une fois de manière asynchrone, où les valeurs finales pour l'index et la longueur sont définies, `history.state` reste inchangée et aucun événement n'est déclenché.

9. Faites défiler jusqu'au fragment indiqué dans le document actif de l'élément navigable .

Si le défilement échoue parce que le `Document` est nouveau et que l'`ID` pertinent n'a pas encore été analysé, le deuxième appel asynchrone pour mettre à jour le document pour l'application de l'étape d'historique se chargera du défilement.

10. Soit *traversable* le *navigable* traversable de navigable .
11. Ajoutez les étapes de navigation synchrone suivantes de l'historique de session impliquant *navigable* à *traversable* :
  1. Finaliser une navigation dans le même document avec *traversable* , *navigable* , *historyEntry* et *entryToReplace* .
  2. Appelez le fragment WebDriver BiDi navigué avec le contexte de navigation actif de *navigable* et un nouveau statut de navigation WebDriver BiDi dont l'`id` est *navigationId* , l'`url` est l'`url` de la ressource et le `statut` est " ".`complete`

Pour **finaliser une navigation dans le même document** avec un traversable navigable *traversable* , un navigable *targetNavigable* , une entrée d'historique de session *targetEntry* et une entrée d'historique de session -ou-null *entryToReplace* :

Ceci est utilisé à la fois par les navigations de fragment et par les étapes de mise à jour de l'URL et de l'historique , qui sont les seules mises à jour synchrones de l'historique de session. Du fait qu'ils sont synchrones, ces algorithmes sont exécutés en dehors de la file d'attente de traversée de l'historique de session de la traversée de niveau supérieur . Cela les désynchronise avec l'étape d'historique de la session en cours du traversable de niveau supérieur . Cet algorithme est donc utilisé pour résoudre les conflits dus aux conditions de concurrence.

1. Assert : cela s'exécute sur la file d'attente de traversée de l'historique de session de *traversable* .
2. Si l'entrée d'historique de session active de *targetNavigable* n'est pas *targetEntry* , alors retournez.
3. Laissez *targetStep* être nul.

4. Laissez *targetEntries* être le résultat de [l'obtention des entrées d'historique de session](#) pour *targetNavigable* .
5. Si *entryToReplace* est nul, alors :
  1. [Effacez l'historique de session avant](#) de *traversable* .
  2. Définissez *targetStep* sur l'étape d'historique de la [session actuelle](#) de *traversable* + 1.
  3. Définissez [le pas](#) de *targetEntry* sur *targetStep* .
  4. [Ajoutez](#) *targetEntry* à *targetEntries* .

Sinon:

5. [Remplacez](#) *entryToReplace* par *targetEntry* dans *targetEntries* .
6. Définissez le [pas](#) de *targetEntry* sur [le pas](#) de *entryToReplace* .
7. Définissez *targetStep* sur [l'étape d'historique de la session en cours](#) de *traversable* .
6. [Appliquez l'étape d'historique](#) *targetStep* à *traversable* .

*Cela se fait même pour [replace](#) les navigations " " , car cela résout les conditions de concurrence sur plusieurs navigations synchrones.*

#### 7.4.2.3.4 Schémas de non-extraction et logiciels externes

Une entrée pour [tenter de créer un document de schéma de non-récupération](#) est la **structure des paramètres de navigation du schéma de non-récupération** . Il s'agit d'une version allégée des [paramètres de navigation](#) qui ne contient que des paramètres pertinents pour le cas de navigation [du schéma sans récupération](#) . Il comporte les [éléments](#) suivants :

##### **origine de l'initiateur**

une [origine](#) pouvant éventuellement être utilisée dans une invite destinée à l'utilisateur pour confirmer l'invocation d'un progiciel externe

*Cela diffère légèrement de l' [origine](#) de l'initiateur d' [un état de document](#) en ce que l' [origine](#) de l'initiateur d'un paramètre de navigation de schéma de non-récupération suit les redirections jusqu'à la dernière URL de schéma de récupération dans une chaîne de redirection qui se termine par une URL de [schéma de non-récupération](#) .*

Pour **tenter de créer un document de schéma de non-récupération** , étant donné une [URL](#) *url* , un [navigable](#) *navigable* , un [ensemble d'indicateurs de](#)



[sandboxing](#) `sandboxFlags` , un [ID de navigation](#) `navigationId` ,  
un [NavigationTimingType](#) `navTimingType` , un booléen `hasTransientActivation` et  
un [origin](#) `initiatorOrigin` :

1. Si `url` doit être gérée à l'aide d'un mécanisme qui n'affecte pas `navigable` , par exemple, parce que [le schéma](#) de `url` est géré en externe, alors :
  1. [Transfert vers un logiciel externe](#) donné `url` , `navigable` , `sandboxFlags` , `hasTransientActivation` et `initiatorOrigin` .
  2. Renvoi nul.
2. Gérer l' `url` en affichant une sorte de contenu en ligne, par exemple, un message d'erreur parce que le schéma spécifié n'est pas l'un des protocoles pris en charge, ou une invite en ligne pour permettre à l'utilisateur de sélectionner [un gestionnaire enregistré](#) pour le schéma donné. Renvoie le résultat de [l'affichage du contenu en ligne](#) donné `navigable` , `navigationId` et `navTimingType` .

*Dans le cas où un gestionnaire enregistré est utilisé, [la navigation](#) sera invoquée avec une nouvelle URL.*

Pour **transférer à un logiciel externe** une [URL](#) ou une ressource [de réponse](#) , un [navigable](#) `navigable` , un [ensemble d'indicateurs de sandboxing](#) `sandboxFlags` , un booléen `hasTransientActivation` et un [origin](#) `initiatorOrigin` , les agents utilisateurs doivent :

1. Si toutes les conditions suivantes sont remplies :
  - `navigable` n'est pas un [traversable de niveau supérieur](#) ;
  - `sandboxFlags` a son ensemble [d'indicateurs de contexte de navigation de navigation de protocoles personnalisés en bac à sable](#) ; et
  - `sandboxFlags` a sa [navigation de niveau supérieur en bac à sable avec l'indicateur de contexte de navigation d'activation de l'utilisateur](#) défini, ou `hasTransientActivation` est faux

puis revenez sans appeler le progiciel externe.

*La navigation à l'intérieur d'une `iframe` vers un logiciel externe peut être vue par les utilisateurs comme une nouvelle fenêtre contextuelle ou une nouvelle navigation de niveau supérieur. C'est pourquoi il n'est autorisé dans le bac à sable `iframe` que lorsque l'un des éléments `allow-popups`, `allow-top-navigation`, `allow-top-navigation-by-user-activation` ou `allow-top-navigation-to-custom-protocols` est spécifié.*

2. Effectuez le transfert approprié de ressource tout en essayant d'atténuer le risque qu'il s'agisse d'une tentative d'exploitation du logiciel cible. Par exemple, les agents utilisateurs pourraient inviter l'utilisateur à confirmer



que *initiatorOrigin* est autorisé à invoquer le logiciel externe en question. En particulier, si *hasTransientActivation* est faux, l'agent utilisateur ne doit pas invoquer le progiciel externe sans confirmation préalable de l'utilisateur.

Par exemple, il pourrait y avoir une vulnérabilité dans le gestionnaire d'URL du logiciel cible qu'une page hostile tenterait d'exploiter en incitant un utilisateur à cliquer sur un lien.

#### 7.4.2.4 Empêcher la navigation

Quelques scénarios peuvent intervenir tôt dans le processus de navigation et mettre le tout à l'arrêt. Cela peut être particulièrement excitant lorsque plusieurs [navigables](#) naviguent en même temps, en raison d'une traversée de l'historique de session.

Une *source* [navigable](#) est **autorisée par le sandboxing à naviguer vers** une seconde *cible* [navigable](#), étant donné [les paramètres d'un instantané source](#) *sourceSnapshotParams*, si les étapes suivantes renvoient true :

1. Si *source* est *target*, alors retourne true.
2. Si *source* est un ancêtre de *target*, alors retourne true.
3. Si *cible* est un ancêtre de *source*, alors :
  1. Si *target* n'est pas un [traversable de niveau supérieur](#), alors retourne true.
  2. Si *sourceSnapshotParams* a [une activation transitoire](#) est true, et que [la navigation de niveau supérieur en bac à sable de sourceSnapshotParams](#)'s sandboxing [flags](#) avec l'indicateur de contexte de navigation d'activation de l'utilisateur est définie, alors renvoie false.
  3. Si *sourceSnapshotParams* a [une activation transitoire](#) est false et que [l'indicateur de navigation de niveau supérieur en bac à sable de sourceSnapshotParams](#) [sandboxing flags](#) sans activation de l'utilisateur est défini, renvoie false.
  4. Renvoie vrai.
4. Si *la cible* est un [traversable de niveau supérieur](#) :
  1. Si *source* est le [seul navigateur sandbox autorisé](#) de *target*, alors renvoie true.

2. Si [l'indicateur de contexte de navigation de navigation en bac à sable](#) des [indicateurs de sandboxing](#) de *sourceSnapshotParams* est défini, renvoie false.
3. Renvoie vrai.
5. Si [l'indicateur de contexte de navigation de navigation en bac à sable](#) des [indicateurs de sandboxing](#) de *sourceSnapshotParams* est défini, renvoie false.
6. Renvoie vrai.

Pour **vérifier si le déchargement est annulé par l'utilisateur** pour [la liste](#) des *navigables* [navigables](#) :

1. Laissez *documents* être le [document actif](#) de chaque [élément](#) dans *navigables* .
2. Laissez *unloadPromptShown* être faux.
3. Laissez *unloadPromptCanceled* être faux.
4. Soit *totalTasks* la [taille](#) des *documents* .
5. Laissez *les tâches terminées* être 0.
6. [Pour chaque](#) *document* de *documents* , [mettez en file d'attente une tâche globale](#) sur la [source de la tâche de navigation et de parcours](#) en fonction de [l'objet global pertinent](#) du *document* pour exécuter les étapes :
  1. Augmentez le [compteur de déchargement](#) du *document* de 1.
  2. Augmentez le [niveau d'imbrication de terminaison](#) de [la boucle d'événements](#) de 1.
  3. Soit *événement* le résultat de [la création d'un événement](#) à l'aide de [BeforeUnloadEvent](#).
  4. Initialise l'attribut de *l'événement* [type](#) à [beforeunload](#) et son [cancelable](#) attribut true.
  5. [Envoyer](#) *l'événement* à [l'objet global pertinent](#) du *document* .
  6. Diminuez le [niveau d'imbrication de terminaison](#) de [la boucle d'événements](#) de 1.
  7. Si toutes les conditions suivantes sont vraies :
    - *unloadPromptShown* est faux ;
    - [l'ensemble d'indicateurs de sandboxing actif](#) du *document* n'a pas son indicateur [de mode sandbox](#) défini ;
    - [l'objet global pertinent](#) du *document* a une [activation persistante](#) ;

- [l'indicateur d'annulation](#) de *l'événement* est défini ou l'[returnValue](#)attribut de *l'événement* n'est pas la chaîne vide ; et
- afficher une invite de déchargement ne sera probablement pas ennuyeux, trompeur ou inutile

alors:

- Définissez *unloadPromptShown* sur true.
- Appelez [l'invite utilisateur WebDriver BiDi ouverte](#) avec [l'objet global pertinent](#) du *document* , " " et `""beforeunload`
- Demandez à l'utilisateur de confirmer qu'il souhaite décharger le document et [faites une pause](#) en attendant la réponse de l'utilisateur.

*Le message affiché à l'utilisateur n'est pas personnalisable, mais plutôt déterminé par l'agent utilisateur. En particulier, la valeur réelle de l'[returnValue](#)attribut est ignorée.*

- Si l'utilisateur n'a pas confirmé la navigation dans la page, définissez *unloadPromptCanceled* sur true.
- Appelez [l'invite utilisateur WebDriver BiDi fermée](#) avec [l'objet global pertinent](#) du *document* et true si *unloadPromptCanceled* est false ou false sinon.

8. Diminue le [compteur de déchargement](#) du *document* de 1.

9. Incrémenter *les tâches terminées* .

7. Attendez que *completeTasks* soit *totalTasks* .

8. Renvoie *unloadPromptCanceled* .

### 7.4.3 Rechargement et déplacement

Pour **recharger** un [navigable](#) *navigable* :

1. Définissez [le rechargement en attente](#) de l'état du [document](#) de [l'entrée active de l'historique de session](#) de *navigable* sur vrai.
2. Soit *traversable* le *navigable* traversable [de navigable](#) .
3. [Ajoutez les étapes de parcours d'historique de session suivantes](#) à *traversable* :
  1. [Appliquez les modifications d'historique en attente](#) à *traversable* avec true.

*Il est intentionnel que l'appel résultant pour appliquer l'étape d'historique ne transmette pas sourceSnapshotParams ou initiatorToCheck . Le rechargement est toujours traité comme s'il était effectué par navigable lui-même, même dans des cas comme `parent.location.reload()` .*

Pour **parcourir l'historique par un delta** étant donné un traversable navigable *traversable* , un entier *delta* et un Document *sourceDocument* facultatif :

1. Laissez *sourceSnapshotParams* et *initiatorToCheck* être nuls.
2. Si *sourceDocument* est donné, alors :
  1. Définissez *sourceSnapshotParams* sur le résultat de l'instantané des paramètres de l'instantané source donné *sourceDocument* .
  2. Définissez *initiatorToCheck* sur le nœud navigable de *sourceDocument* .
3. Ajoutez les étapes de parcours d'historique de session suivantes à *traversable* :
  1. Soit *allSteps* le résultat de l'obtention de toutes les étapes d'historique utilisées pour *traversable* .
  2. Soit *currentStepIndex* l'index de l'étape d'historique de la session en cours de *traversable* dans *allSteps* .
  3. Soit *targetStepIndex* soit *currentStepIndex* plus *delta* .
  4. Si *allSteps* [ *targetStepIndex* ] n'existe pas , abandonnez ces étapes.
  5. Appliquez l'étape d'historique *allSteps* [ *targetStepIndex* ] à *traversable* , avec checkForUserCancelation défini sur *true*, sourceSnapshotParams défini sur *sourceSnapshotParams* et initiatorToCheck défini sur *initiatorToCheck* .

#### 7.4.4 "Navigations" synchrones sans fragment

Outre l' algorithme de navigation , les entrées de l'historique de session peuvent être poussées ou remplacées via un autre mécanisme, les étapes de mise à jour de l'URL et de l'historique . Les appelants les plus connus de ces étapes sont les API history.replaceState() et history.pushState() , mais diverses autres parties de la norme doivent également effectuer des mises à jour de l' entrée d'historique active , et elles utilisent ces étapes pour le faire.

Les **étapes de mise à jour de l'URL et de l'historique**, étant donné un Document *document*, une URL *newURL*, un état sérialisé facultatif -or- null ***serializedData*** (null par défaut) et un comportement facultatif de gestion de l'historique ***historyHandling*** (par défaut "replace"), sont :

1. Soit *navigable* le nœud navigable du *document*.
2. Soit *activeEntry* l'entrée d'historique de session active de *navigable*.
3. Soit *newEntry* une nouvelle entrée d'historique de session, avec

URL

*nouvelleURL*

état sérialisé

si *serializedData* n'est pas null, *serializedData* ; sinon l'état sérialisé de *activeEntry*

état des documents

État du document de *activeEntry*

défilement mode de restauration

Mode de restauration du défilement de *activeEntry*

état utilisateur persistant

État utilisateur persistant de *activeEntry*

4. Si le *document* est initial*about:blank* est vrai, alors définissez *historyHandling* sur "replace".

*Cela signifie que pushState() sur une initiale*about:blank* Document se comporte comme un replaceState() appel.*

5. Soit *entryToReplace* être *activeEntry* si *historyHandling* est "replace", sinon null.
6. Si *historyHandling* vaut "push", alors :
  1. Incrémente l'index de l' objet historique du *document*.
  2. Définissez la longueur de l'objet historique du *document* sur son index + 1.

*Il s'agit de valeurs de meilleure estimation temporaires pour un accès synchrone immédiat.*

7. Si *serializedData* n'est pas null, alors restaurez l'état de l'objet historique donné *document* et *newEntry*.
8. Définissez l'URL du *document* sur *newURL*.

*Comme il ne s'agit ni d'une navigation ni d'une traversée d'historique, cela ne provoque pas hashchange le déclenchement d'un événement.*

9. Définissez la dernière entrée du *document* sur *newEntry*.

10. Définissez [l'entrée de l'historique des sessions actives](#) de *navigable* sur *newEntry* .
11. Soit *traversable* le *navigable* traversable [de navigable](#) .
12. [Ajoutez les étapes de navigation synchrone suivantes de l'historique de session](#) impliquant *navigable* à *traversable* :
  1. [Finaliser une navigation dans le même document](#) avec *traversable* , *navigable* , *newEntry* et *entryToReplace* .

*Bien que [la navigation par fragment](#) et [les étapes de mise à jour d'URL et d'historique](#) effectuent des mises à jour d'historique synchrones, seule la navigation par fragment contient un appel synchrone pour [mettre à jour le document pour l'application de l'étape d'historique](#) . Les étapes de mise à jour de l'URL et de l'historique effectuent à la place quelques mises à jour sélectionnées dans l'algorithme ci-dessus, en omettant les autres. Il s'agit en quelque sorte d'un accident historique malheureux, et conduit généralement à la [tristesse des développeurs Web](#) face à l'incohérence. Par exemple, cela signifie que `popstate` les événements se déclenchent pour les navigations de fragments, mais pas pour `history.pushState()` les appels.*

#### 7.4.5 Remplir une entrée d'historique de session

Comme expliqué dans [la vue d'ensemble](#) , [la navigation](#) et [la traversée](#) impliquent la création d'une [entrée d'historique de session](#) , puis la tentative de remplissage de son membre [de document](#) , afin qu'il puisse être présenté à l'intérieur du [navigable](#) .

Cela implique soit : d'utiliser [une réponse déjà donnée](#) ; en utilisant la [ressource srcdoc](#) stockée dans l' [entrée de l'historique de session](#) ; ou [aller chercher](#) . Le processus a plusieurs modes d'échec, qui peuvent soit ne rien faire (laisser le [navigable](#) sur son état [actif Document](#) ) soit remplir l' [entrée de l'historique de session](#) avec un [document d'erreur](#) .

Pour **tenter de remplir le document de l'entrée d'historique** pour une entrée [d'historique de session](#) , étant donné un *navigable* [navigable](#) , un *navTimingType* , un [snapshot source params](#) *sourceSnapshotParams* , un [snapshot cible params](#) *targetSnapshotParams* , un [ID de navigation](#) facultatif -ou- null *navigationId* (null par défaut), un [params de navigation](#) facultatifs -or- null *navigationParams* (null par défaut), une chaîne facultative *cspNavigationType* (par défaut " "), un booléen facultatif ***allowPOST*** [NavigationTimingType](#) *other* (false par défaut) et les étapes facultatives de l'algorithme ***CompletionSteps*** (par défaut, un algorithme vide) :

1. [Assert](#) : cela fonctionne [en parallèle](#) .

2. [Assert](#) : si *navigationParams* est non nul, alors [la réponse](#) de *navigationParams* est non nulle.
3. Soit *currentBrowsingContext* le [contexte de navigation actif](#) de *navigable* .
4. Soit *documentResource* la [ressource](#) de l' état du [document de l'entrée](#) .
5. Si *navigationParams* est nul, alors :
  1. Si *documentResource* est une chaîne, définissez *navigationParams* sur le résultat de [la création de paramètres de navigation à partir d'une ressource srcdoc](#) donnée *entry* , *navigable* , *targetSnapshotParams* , *navigationId* et *navTimingType* .
  2. Sinon, si les deux conditions suivantes sont vraies :
    - [le schéma](#) de l' [URL](#) de l' *entrée* est un [schéma de récupération](#) ; et
    - *documentResource* est null, ou *allowPOST* est vrai et [le corps](#) de la requête de *documentResource* n'est pas un échec

puis définissez *navigationParams* sur le résultat de [la création des paramètres de navigation en récupérant l'entrée](#) donnée , *navigable* , *sourceSnapshotParams* , *targetSnapshotParams* , *cspNavigationType* , *navigationId* et *navTimingType* .
  3. Sinon, si [le schéma](#) de l'[URL](#) de l'*entrée* n'est pas un [schéma de récupération](#) , définissez *navigationParams* sur un nouveau [schéma de navigation params non-récupération](#) , avec  
[origine de l'initiateur](#)  
[origine](#) de l'initiateur de l'état du [document](#) de l'*entrée*
6. [Mettez une tâche globale en file d'attente](#) sur la [source de la tâche de navigation et de traversée](#) , en fonction de [la fenêtre active](#) de *navigable* , pour exécuter ces étapes :
  1. Si [la navigation en cours](#) de *navigable* n'est plus égale à *navigationId* , alors exécutez *CompletionSteps* et revenez.
  2. Que l'échec soit faux.
  3. Si *navigationParams* est un [paramètre de navigation de schéma de non-récupération](#) , alors définissez l'[état](#) du [document](#) de l' *entrée* sur le résultat de l'exécution de [la tentative de création d'un document de schéma de non-récupération](#) étant donné l'[URL](#) de l'*entrée* , *navigable* , [les indicateurs de sandboxing](#) de *targetSnapshotParams* , *navigationId* , *navTimingType* ,



*sourceSnapshotParams* a [une activation transitoire](#) et [l'origine](#) de l'initiateur de *navigationParams* .

*L'URL de l'entrée peut avoir été modifiée au cours de l'étape précédente de cet algorithme suite à une redirection HTTP.*

4. Sinon, si *navigationParams* est null, définissez l'échec sur true.
5. Sinon, si le résultat de [la réponse de navigation à la requête de navigation de type dans la cible doit-il être bloqué par la politique de sécurité du contenu ?](#) étant donné [la demande](#) de *navigationParams* , [la réponse](#) de *navigationParams* , [la liste CSP](#) du [conteneur de stratégie](#) de *navigationParams* , *cspNavigationType* et *navigable* est " " , puis définissez l' échec sur true. `[CSP]Blocked`
6. Sinon, si [l'environnement réservé](#) de *navigationParams* n'est pas nul et que le résultat de [la vérification de la conformité d'une réponse de navigation à sa stratégie d'intégration](#) étant donné [la réponse](#) de *navigationParams* , *navigable* et [la stratégie d'intégration](#) du [conteneur de stratégie de](#) *navigationParams* est faux, définissez l'échec sur vrai.
7. Sinon, si le résultat de [la vérification de l'adhésion d'une réponse de navigation à `X-Frame-Options` la réponse](#) donnée de *navigationParams* , *navigable* , [la liste CSP](#) du [conteneur de politique](#) de *navigationParams* et [l'origine](#) de *navigationParams* est faux, alors définissez l'échec sur vrai.
8. Si l'échec est vrai, alors :
  - Définissez le [document de l'état](#) du document de l'entrée sur le résultat de [la création d'un document pour le contenu en ligne qui n'a pas de DOM](#) , étant donné *navigable* , null et *navTimingType* . Le contenu en ligne doit indiquer à l'utilisateur le type d'erreur qui s'est produit.
  - Définissez l'état du [document](#) de [l'entrée](#) sur [le document](#) récupérable sur false.
  - Si *navigationParams* n'est pas nul, alors :
    1. Exécutez l' [environnement en ignorant les étapes](#) pour [l'environnement réservé](#) de *navigationParams* .
    2. L'appel [de la navigation WebDriver BiDi a échoué](#) avec *currentBrowsingContext* et un nouveau [statut de navigation WebDriver BiDi](#) dont [l'id](#) est *navigationId* , [le statut](#) est " `canceled` " et [l'url](#) est l' [URL de la réponse](#) de *navigationParams* .



9. Sinon, si [le statut](#) de [la réponse](#) de *navigationParams* est 204 ou 205, alors :

- Exécutez *les étapes de complétion* .
- Retour.

10. Sinon, si [la réponse](#) de *navigationParams* a un en-tête `` spécifiant le type de disposition, alors :[Content-Disposition](#)*attachment*

- Laissez *sourceAllowsDownloading* être *sourceSnapshotParams* 's [allow downloading](#) .
- Laissez *targetAllowsDownloading* être false si [l'indicateur de sandboxing final](#) de *navigationParams* a l' [indicateur de contexte de navigation des téléchargements en sandbox](#) défini ; autrement vrai.
- Si le résultat de l'exécution [de l'autorisation de téléchargement](#) avec *sourceAllowsDownloading* et *targetAllowsDownloading* est vrai, traitez [la réponse](#) de *navigationParams* [comme un download](#) .
- Appelez [le téléchargement de WebDriver BiDi démarré](#) avec *currentBrowsingContext* et un nouveau [statut de navigation WebDriver BiDi](#) dont [l'id](#) est *navigationId* , [le statut](#) est "[complete](#)" et [l' url](#) est [l' URL](#) de [la réponse](#) de *navigationParams* .
- Exécutez *les étapes de complétion* .
- Retour.

11. Sinon:

- Soit *document* le résultat du [chargement d'un document](#) donné par *navigationParams* , *sourceSnapshotParams* et l'origine de [l'initiateur](#) de l'état du [document](#) de l'entrée .
- Si *le document* est nul, exécutez *les étapes de complétion* et revenez.
- Définissez [l'état](#) du [document](#) de l'entrée sur *document* .
- Définissez [l'origine](#) de l' état du [document de l'entrée](#) sur [l'origine](#) du *document* .
- Si [l'URL](#) du *document* [requiert le stockage du conteneur de politique dans l'historique](#) , définissez le [conteneur de politique historique](#) de l'état du [document de l'entrée](#) sur [le conteneur de politique](#) de *navigationParams* .

12. Si le référent de la requête de l'état du document de *l'entrée* est " ", alors définissez-le sur le référent de la *requête* `.client`

*Cela garantit que si nous traversons une entrée arrière et que nous devons récupérer à nouveau, nous utilisons le même référent, au lieu de dériver le référent du client de récupération.*

13. Si le document de l'état du document de *l'entrée* n'est pas nul, définissez l'état du document de *l'entrée* toujours rempli sur vrai.

14. Exécutez les étapes de complétion .

Pour **créer des paramètres de navigation à partir d'une ressource srcdoc** en fonction d'une *entrée* d'historique de session entry , d'un navigable *navigable* , d'un snapshot cible params *targetSnapshotParams* , d'un ID de navigation -ou- null *navigationId* et d'un *navTimingType* : NavigationTimingType

1. Soit *documentResource* la ressource de l'état du document de *l'entrée* .
2. Soit *réponse* une nouvelle réponse avec

URL

`about:srcdoc`

liste d'en-tête

« ( Content-Type , ` text/html ` ) »

corps

l'encodage UTF-8 de *documentResource* , en tant que corps

3. Soit *responseOrigin* le résultat de la détermination de l'origine en fonction de l'URL de la réponse , des indicateurs de sandboxing de *targetSnapshotParams* , null et de l'origine de l'état du document de *l'entrée* .
4. Laissez *coop* être une nouvelle politique d'ouverture d'origine croisée .
5. Soit *coopEnforcementResult* un nouveau résultat d'application de la politique d'ouverture d'origine croisée avec

URL

URL de la réponse

origine

*réponseOrigin*

politique d'ouverture d'origine croisée

*coopérative*

6. Soit *policyContainer* le résultat de la détermination des paramètres de navigation du conteneur de stratégie en fonction de l'URL de la réponse , du conteneur de stratégie de l'historique de l'état du document de *l'entrée* , de la valeur null, du conteneur de stratégie du document *navigable* et de la valeur null.
7. Renvoie un nouveau paramètre de navigation , avec

identifiant

*ID de navigation*

demande

*nul*

réponse

*réponse*

origine

*réponseOrigine*

conteneur de stratégie

*policyContainer*

ensemble final de drapeaux de sandboxing

*Drapeaux de sandboxing de targetSnapshotParams*

politique d'ouverture d'origine croisée

*coopérative*

Résultat de l'application COOP

*coopEnforcementResult*

environnement réservé

*nul*

navigable

*navigable*

type de temps de navigation

*navTimingType*

récupérer le contrôleur

*nul*

valider les premiers indices

*nul*

Pour **créer des paramètres de navigation en récupérant** une *entrée* d'historique de session donnée , un navigable *navigable* , des paramètres d'instantané source *sourceSnapshotParams* , des paramètres d'instantané cible *targetSnapshotParams* , une chaîne *cspNavigationType* , un ID de navigation - ou-null *navigationId* et un *navTimingType* , procédez comme suit . Ils renvoient un paramètre de navigation , un schéma de navigation non-fetch params ou null. *NavigationTimingType*

*Cet algorithme mute entry .*

1. Assert : cela fonctionne en parallèle .
2. Soit *documentResource* la ressource de l' état du document de l'entrée .
3. Soit *request* une nouvelle requête , avec

URL

URL de l'entrée

client

Client de récupération de *sourceSnapshotParams*

destination

*" document "*

mode d'identification

"include"

utiliser l'indicateur d'informations d'identification d'URL

ensemble

mode de redirection

"manual"

remplace l'identifiant client

les paramètres pertinents du document actif de *navigable* l'identifiant de l'objet

mode

"navigate"

réfèrent

réfèrent de la requête de l'état du document de l'entrée

politique de référence

politique de référence de la demande de l'état du document de l'entrée

4. Si *documentResource* est une ressource POST , alors :
  1. Définissez la méthode de *la requête* sur ``POST`
  2. Définissez le corps de *la requête* sur le corps de la requête de *documentResource* .
  3. Définissez ``Content-Type`` sur le type de contenu de la requête de *documentResource* dans la liste d'en-tête de la *requête* .
5. Si le rechargement en attente de l'état du document de l'entrée est vrai, alors définissez l'indicateur de rechargement-navigation de la *requête* .
6. Sinon, si l'état du document de l'entrée n'a jamais été rempli est vrai, alors définissez l'indicateur de navigation de l' historique de la *requête* .
7. Si l'activation transitoire de *sourceSnapshotParams* est true, alors définissez l'activation de l'utilisateur de *request* sur true.
8. Si le conteneur de *navigable* n'est pas nul :
  1. Si le conteneur de l' élément *navigable* a une origine de portée de contexte de navigation , alors définissez l'origine de la *requête* sur cette origine de portée de contexte de navigation .
  2. Définissez la destination de la *requête* sur le nom local du conteneur *navigable* .
  3. Si le client de récupération de *sourceSnapshotParams* est l'objet de paramètres pertinent du document conteneur de *navigable* , définissez le type d' initiateur de la *demande* sur le nom local du conteneur de *navigable* .

*Cela garantit que seules les navigations initiées par le conteneur sont signalées à la synchronisation des ressources.*
9. Soit *la réponse* nulle.

10. Laissez *responseOrigin* être null.
11. Laissez *fetchController* être nul.
12. Soit *coopEnforcementResult* un nouveau [résultat d'application de la politique d'ouverture d'origine croisée](#) , avec  
[URL](#)  
[URL](#) du [document actif](#) de *navigable*  
[origine](#)  
[origine](#) du [document actif](#) de *navigable*  
[politique d'ouverture d'origine croisée](#)  
[politique d'ouverture croisée](#) du [document actif](#) de *navigable*  
[le contexte actuel est la source de navigation](#)  
true si l' [origine](#) du [document actif](#) de *navigable* est [la même origine](#) que l'[origine](#) de l'initiateur de l'état du [document de](#) l'entrée sinon false
13. Soit *finalSandboxFlags* un [ensemble d'indicateurs de sandboxing vide](#) .
14. Laissez *responsePolicyContainer* être nul.
15. Soit *responseCOOP* une nouvelle [politique d'ouverture cross-origin](#) .
16. Laissez *locationURL* être nul.
17. Soit *currentURL* l' [URL actuelle](#) de la requête .
18. Laissez *commitEarlyHints* être nul.
19. Alors que c'est vrai :

1. Si [le client réservé](#) de la requête n'est pas nul et que l'[origine](#) de l'[URL actuelle](#) n'est pas la [même](#) que l' [origine](#) de l' [URL de création](#) du [client réservé](#) de la requête , alors :

1. Exécutez l' [environnement en ignorant les étapes](#) pour [le client réservé](#) de la requête .
2. Définissez [le client réservé](#) de la demande sur null.
3. Définissez *commitEarlyHints* sur null.

*Les liens préchargés des [premiers en-têtes d'indication](#) restent dans le cache de préchargement après une [même](#) redirection d'origine, mais sont supprimés lorsque la redirection est d'origine croisée.*

2. Si [le client réservé](#) de request est nul, alors :
  1. Soit *topLevelCreationURL* soit *currentURL* .
  2. Laissez *topLevelOrigin* être nul.

3. Si *navigable* n'est pas un [traversable de niveau supérieur](#) , alors :
  1. Soit *parentEnvironment* l'objet de [paramètres pertinent](#) du [document actif](#) parent de *navigable* .
  2. Définissez *topLevelCreationURL* sur l' [URL de création de niveau supérieur](#) de *parentEnvironment* .
  3. Définissez *topLevelOrigin* sur l'[origine de niveau supérieur](#) de *parentEnvironment* .
4. Définissez [le client réservé](#) de la requête sur un nouvel [environnement](#) dont l'[ID](#) est une chaîne opaque unique, [le contexte de navigation cible](#) est [le contexte de navigation actif](#) de *navigable* , l'[URL de création](#) est *currentURL* , l'[URL de création de niveau supérieur](#) est *topLevelCreationURL* et l'[origine de niveau supérieur](#) est *topLevelOrigin* .

*L'agent de service actif de l'environnement créé est défini dans l'algorithme [d'extraction de poignée](#) lors de l'extraction si l'URL de la demande correspond à un enregistrement d'agent de service. [SW]*

3. Si le résultat de [la demande de navigation de type doit être bloqué par la politique de sécurité du contenu ?](#) requête donnée et *cspNavigationType* est "Blocked", puis définissez *réponse* à une [erreur réseau](#) et [break](#) . [CSP]
4. Définissez *la réponse* sur null.
5. Si *fetchController* est null, définissez *fetchController* sur le résultat de [la récupération](#) de *request* , avec [processEarlyHintsResponse](#) défini sur *processEarlyHintsResponse* comme défini ci-dessous, [processResponse](#) défini sur *processResponse* comme défini ci-dessous et [useParallelQueue](#) défini sur true.

Soit *processEarlyHintsResponse* l'algorithme suivant étant donné une [réponse](#) *earlyResponse* :

1. Si *commitEarlyHints* a la valeur null, définissez *commitEarlyHints* sur le résultat du [traitement des premiers en-têtes d'indice](#) donnés par *earlyResponse* et [le client réservé](#) de la requête .

Soit *processResponse* l'algorithme suivant étant donné une [réponse](#) *fetchedResponse* :

2. Définissez *la réponse* sur *fetchedResponse* .
6. Sinon, [traitez la prochaine redirection manuelle](#) pour *fetchController* .

Cela se traduira par l'appel du [processResponse](#) que nous avons fourni ci-dessus, lors de notre première itération dans la boucle, et donc la définition de réponse .  
La navigation gère les redirections manuellement car la navigation est le seul endroit de la plate-forme Web qui s'occupe des redirections vers [mailto:](#) les URL et autres.

7. Attendez que l'une des réponses ne soit pas nulle ou que la [navigation en cours](#) de [navigable](#) ne soit plus égale à [navigationId](#) .

Si cette dernière condition se produit, [abandonnez](#) [fetchController](#) et revenez.

Sinon, continuez.

8. Si [le corps](#) de la *demande* est nul, définissez [la ressource](#) de l'état du [document de](#) l'entrée sur nul.

*Fetch annule la configuration du [corps](#) pour des redirections particulières.*

9. Définissez *responsePolicyContainer* sur le résultat de [la création d'un conteneur de règles à partir d'une réponse d'extraction](#) en fonction de la réponse et du [client réservé](#) de la requête .
10. Définissez *finalSandboxFlags* sur l' [union](#) des [indicateurs de sandboxing](#) de *targetSnapshotParams* et [des indicateurs de sandboxing dérivés](#) du [CSP de la liste](#) CSP de *responsePolicyContainer* .
11. Définissez *responseOrigin* sur le résultat de [la détermination de l'origine](#) de l' [URL](#) de la réponse , des *finalSandboxFlags* , de l'origine de [l'initiateur](#) de l'état du [document de](#) l'entrée et de null.
12. Si *navigable* est un [traversable de niveau supérieur](#) , alors :

1. Définissez *responseCOOP* sur le résultat de [l'obtention d'une politique d'ouverture cross-origin](#) compte tenu de la réponse et du [client réservé](#) de la requête .
2. Définissez *coopEnforcementResult* sur le résultat de [l'application de la politique d'ouverture d'origine croisée de la réponse](#) en fonction [du contexte de navigation actif](#) de *navigable* , de l' [URL](#) de la réponse , de *responseOrigin* , de *responseCOOP* , de *coopEnforcementResult* et [du référent](#) de la demande .
3. Si *finalSandboxFlags* n'est pas vide et que [la valeur](#) de *responseCOOP* n'est pas " " , alors définissez *response* sur une [erreur réseau](#) appropriée et [break](#) [.unsafe-none](#)



*Cela entraîne une erreur de réseau car il est impossible de fournir simultanément une table rase à une réponse à l'aide de la politique d'ouverture d'origine croisée et du bac à sable le résultat de la navigation vers cette réponse.*

13. Si la réponse n'est pas une [erreur réseau](#) , *navigable* est un [enfant navigable](#) , et le résultat de l'exécution d'une [vérification de stratégie de ressources d'origine croisée](#) avec l' [origine](#) du [document conteneur](#) de *navigable* , l'[objet de paramètres pertinent](#) du document [conteneur de navigable](#) , la requête [destination](#) , *response* et *true* sont **bloqués** , puis définissez *response* sur une [erreur réseau](#) et [break](#) .

*Ici, nous exécutons la [vérification de la politique de ressources cross-origin](#) par rapport au [navigable parent](#) plutôt qu'au *navigable* lui-même. En effet, nous nous soucions de la même origine du contenu intégré par rapport au contexte parent, et non de la source de navigation.*

14. Définissez *locationURL* sur l'[URL d'emplacement](#) de la réponse en fonction du [fragment](#) de *currentURL* .
15. Si *locationURL* est un échec ou nul, alors [break](#) .
16. [Affirmer](#) : *locationURL* est une [URL](#) .
17. Définissez l'[état sérialisé](#) de l'entrée sur [StructuredSerializeForStorage](#) (null).
18. Soit *oldDocState* l' [état du document](#) de l'entrée .
19. Définir l' [état du document](#) de l' entrée sur un nouvel [état du document](#) , avec

**[conteneur de stratégie d'historique](#)**

un [clone](#) du [conteneur de politique d'historique](#) de *oldDocState* s'il n'est pas nul ; nul sinon

**[réfèrent de la demande](#)**

[réfèrent de requête](#) de *oldDocState*

**[demander la politique de référence](#)**

[politique de référence de la demande](#) de *oldDocState*

**[origine](#)**

[origine](#) de *oldDocState*

**[Ressource](#)**

[ressource](#) de *oldDocState*

**[toujours peuplé](#)**

*oldDocState* n'a [jamais été rempli](#)

**[nom de la cible navigable](#)**

[nom de la cible navigable](#) de *oldDocState*

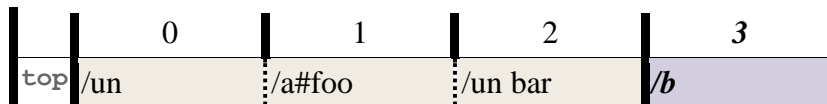
*Pour le cas de la navigation, seule l'entrée faisait référence à *oldDocState* , qui a été créé [au début de l'algorithme de navigation](#) . Donc, pour les navigations, il ne s'agit fonctionnellement*



que d'une mise à jour de l' état du document de l'entrée . Pour le cas de la traversée, il est possible que des entrées d'historique de session adjacentes fassent également référence à `oldDocState` , auquel cas elles continueront à le faire même après que nous ayons mis à jour l' état du document de l' entrée .

Le conteneur de politique d'historique de `oldDocState` n'est jamais non nul ici dans le cas de la traversée, après l'avoir rempli lors d'une navigation vers une URL qui nécessite de stocker le conteneur de politique dans history .

La configuration est donnée par le diagramme de Jake suivant :



Supposons également que l' état du document partagé par les entrées des étapes 0, 1 et 2 ait un document nul , c'est-à-dire que `bfcache` n'est pas en jeu.

Considérons maintenant le scénario où nous revenons à l'étape 2, mais cette fois lors de la récupération de `/a`, le serveur répond avec un `Location`-en-tête `` pointant vers `/c`. Autrement dit, `locationURL` pointe vers `/c` et nous avons donc atteint cette étape au lieu de sortir de la boucle.

Dans ce cas, nous remplaçons l' état de document de l' entrée d'historique de session occupant l'étape 2, mais nous ne remplaçons pas l'état de document des entrées occupant les étapes 0 et 1. Le diagramme de Jake résultant ressemble à ceci :



Notez que nous effectuons ce remplacement même si nous nous retrouvons dans une chaîne de redirection vers l'URL d'origine, par exemple si `/c` elle-même avait un `Location`-en-tête `` pointant vers `/a`. Un tel cas finirait ainsi :



20. Si le schéma de `locationURL` n'est pas un schéma HTTP(S) , alors :

1. Définissez la ressource de l' état du document de l'entrée sur null.
2. Pause .

21. Définissez `currentURL` sur `locationURL` .

22. Définissez l'URL de l'entrée sur `currentURL` .

*À la fin de cette boucle, nous serons dans l'un de ces scénarios :*

23. `locationURL` est un échec, en raison d'un `Location` en-tête `` non analysable.

24. `locationURL` est null, soit parce que la réponse est une erreur réseau , soit parce que nous avons récupéré avec succès une réponse HTTP(S) sans erreur réseau sans `Location` en-tête `` .

25. `locationURL` est une URL avec un schéma non- HTTP(S) .

20. Si `locationURL` est une URL dont le schéma n'est pas un schéma de récupération , renvoie alors un nouveau schéma de navigation params non-récupération , avec

**origine de l'initiateur**

origine de l' URL actuelle de la requête

*À ce stade, l'URL actuelle de la requête est la dernière URL de la chaîne de redirection avec un schéma de récupération avant la redirection vers une URL sans schéma de récupération . C'est l'origine de cette URL qui sera utilisée comme origine de l'initiateur pour les navigations vers les URL de schéma de non-récupération .*

21. Si l'une des conditions suivantes est vraie :

1. la réponse est une erreur réseau ;
2. `locationURL` est un échec ; ou
3. `locationURL` est une URL dont le schéma est un schéma de récupération

puis retourne null.

*Nous autorisons les redirections vers des URL de schéma de non-récupération , mais les redirections vers des URL de schéma de récupération qui ne sont pas HTTP(S) sont traitées comme des erreurs de réseau.*

22. Assert : `locationURL` est null et la réponse n'est pas une erreur réseau .

23. Soit `resultPolicyContainer` le résultat de la détermination du conteneur de stratégie des paramètres de navigation en fonction de l' URL de la réponse , du conteneur de stratégie d'historique de l'état du document de l'entrée , du conteneur de stratégie source de `sourceSnapshotParams` , de null et de `responsePolicyContainer` .

24. Renvoie un nouveau paramètre de navigation , avec

**identifiant**

*ID de navigation*

demande

*demande*

réponse

*réponse*

origine

*réponseOrigine*

conteneur de stratégie

*resultPolicyContainer*

ensemble final de drapeaux de sandboxing

*finalSandboxFlagsfinalSandboxFlags*

politique d'ouverture d'origine croisée

*réponseCOOP*

Résultat de l'application COOP

*coopEnforcementResult*

environnement réservé

client réservé de la requête

navigable

*navigable*

type de temps de navigation

*navTimingType*

recupérer le contrôleur

*fetchController*

valider les premiers indices

*commitEarlyHints*

Un élément a une **origine de portée de contexte de navigation** si le nœud navigable[Document](#) de son est un traversable de niveau supérieur ou si tous ses navigables ancêtres ont tous des documents actifs dont les origines sont la même origine que l'origine du document de nœud de l'élément . Si un élément a une portée de contexte de navigation origin , alors sa valeur est l' origine du nœud document de l'élément .[Document](#)

Cette définition est brisée et nécessite une enquête pour voir ce qu'elle était censée exprimer : voir le numéro 4703 .

Pour **charger un document** avec les paramètres de navigation *navigationParams* , les paramètres d'instantané source *sourceSnapshotParams* et l'initiateur d'origineOrigin , procédez comme suit. Ils renvoient a ou null.[Document](#)

1. Soit *type* le type calculé de la réponse de *navigationParams* .
2. Si l'agent utilisateur a été configuré pour traiter les ressources du *type* donné en utilisant un mécanisme autre que le rendu du contenu dans un navigable , ignorez cette étape. Sinon, si le *type* est l'un des types suivants :

un type HTML MIME

Renvoie le résultat du [chargement d'un document HTML](#) ,  
donné *navigationParams* .

un [type XML MIME](#) qui n'est pas un [type XML MIME explicitement pris en charge](#)

Renvoie le résultat du [chargement d'un document XML](#) avec *navigationParams* et *type* .

un [type MIME JavaScript](#)

un [type JSON MIME](#) qui n'est pas un [type JSON MIME explicitement pris en charge](#)

" [text/css](#) "

" [text/plain](#) "

" [text/vtt](#) "

Renvoie le résultat du [chargement d'un document texte](#) avec *navigationParams* et *type* .

" [multipart/x-mixed-replace](#) "

Renvoie le résultat du [chargement d'un multipart/x-mixed-replace document](#) ,  
étant donné *navigationParams* , *sourceSnapshotParams* et *initiatorOrigin* .

**Un type d'image, de vidéo ou d'audio pris en charge**

Renvoie le résultat du [chargement d'un document multimédia](#) avec *navigationParams* et *type* .

" application/pdf "

" text/pdf "

Si le [visualiseur PDF de l'agent utilisateur pris en charge](#) est true, renvoie le  
résultat de [la création d'un document pour le contenu en ligne qui n'a pas de DOM](#) étant donné [le navigable](#) de *navigationParams* .

Sinon, continuez.

Un **type XML MIME explicitement pris en charge** est un [type XML MIME](#) pour lequel l'agent utilisateur est configuré pour utiliser une application externe pour restituer le contenu, ou pour lequel l'agent utilisateur a des règles de traitement dédiées. Par exemple, un navigateur Web avec un visualiseur de flux Atom intégré prendrait explicitement en charge le [application/atom+xml](#) type MIME.

Un **type JSON MIME explicitement pris en charge** est un [type JSON MIME](#) pour lequel l'agent utilisateur est configuré pour utiliser une application externe pour restituer le contenu, ou pour lequel l'agent utilisateur dispose de règles de traitement dédiées.

*Dans les deux cas, l'application externe ou l'agent utilisateur affichera [le contenu](#) directement dans [le navigable](#) de *navigationParams* ou [le transmettra à un logiciel externe](#) . Les deux se produisent dans les étapes ci-dessous.*

3. Si, étant donné *type*, la nouvelle ressource doit être gérée en affichant une sorte de contenu en ligne, par exemple, un rendu natif du contenu ou un message d'erreur parce que le type spécifié n'est pas pris en charge, alors retournez le résultat de la création d'un document pour en [ligne contenu qui n'a pas de DOM](#) étant donné [le navigable](#) de *navigationParams*, [l'id](#) de *navigationParams* et [le type de synchronisation de navigation](#) de *navigationParams*.
4. Sinon, le *type* du document est tel que la ressource n'affectera pas [le navigable](#) de *navigationParams*, par exemple parce que la ressource doit être transmise à une application externe ou parce qu'il s'agit d'un type inconnu qui sera traité [comme un téléchargement](#). [Transfert vers un logiciel externe](#) en fonction de [la réponse](#) de *navigationParams*, [de navigable](#) de *navigationParams*, de [l'indicateur final de sandboxing](#) de *navigationParams*, de [l'activation transitoire](#) de *sourceSnapshotParams* et de *initiatorOrigin*.
5. Renvoie nul.

## 7.4.6 Application de l'étape d'historique

Pour la navigation et le parcours, une fois que nous avons une idée de l'endroit où nous voulons aller dans l'historique de la session, une grande partie du travail consiste à appliquer cette notion au navigable traversable [et](#) au pertinent [Document](#). Pour les navigations, ce travail intervient généralement vers la fin du processus ; pour les traversées, c'est le début.

### 7.4.6.1 Mise à jour du traversable

S'assurer qu'un [traversable](#) se termine à la bonne étape de l'historique de session est particulièrement complexe, car cela peut impliquer la coordination entre plusieurs descendants [navigables](#) du traversable, leur [remplissage](#) en parallèle, puis la synchronisation de sauvegarde pour s'assurer que tout le monde a la même vue du résultat. Cela est encore compliqué par l'existence de navigations synchrones dans le même document mélangées avec des navigations entre documents, et par la façon dont les pages Web ont fini par avoir certaines attentes relatives en matière de synchronisation.

Un **état de continuation navigable changeant** est utilisé pour stocker des informations pendant l' [application de l'](#) algorithme d'étape d'historique, permettant à des parties de l'algorithme de continuer uniquement après que d'autres parties sont terminées. C'est une [structure](#) avec :

***document affiché***

UNDocument

**entrée cible**

Une [entrée d'historique de session](#)

**navigable**

Une [navigable](#)

**mise à jour uniquement**

Un booléen

Pour **appliquer l'étape d'historique** pas d'entier non négatif à un [traversable](#) [navigable traversable](#), avec un booléen facultatif **checkForUserCancelation** (faux par défaut), [des paramètres d'instantané source](#) facultatifs -ou- null **sourceSnapshotParams** (null par défaut) et **un initiatorToCheck** [navigable](#) facultatif :

*sourceSnapshotParams et initiatorToCheck sont toujours donnés ou non donnés. Ils ne sont généralement pas donnés, car la plupart des appelants n'ont pas besoin des vérifications supplémentaires sur l'initiateur de navigation qu'ils provoquent. (Peut-être parce que l'appelant a déjà effectué ces vérifications lui-même.)*

1. [Assert](#) : Cela s'exécute dans [la file d'attente de traversée](#) de l'historique de session de *traversable* .
2. Soit *targetStep* le résultat de [l'obtention de l'étape utilisée](#) donnée *traversable* et *step* .
3. Si *initiatorToCheck* est donné, alors :
  1. [Assert](#) : *sourceSnapshotParams* n'est pas nul.
  2. [Pour chaque](#) *élément navigable* de [obtenir tous les éléments navigables dont l'entrée d'historique de la session en cours va changer ou se recharger](#) : si *initiatorToCheck* n'est pas [autorisé par le sandboxing à naviguer](#) dans les *éléments navigables* donnés *sourceSnapshotParams* , puis retournez.
4. Laissez *navigablesCrossingDocuments* être le résultat de [l'obtention de tous les éléments navigables susceptibles de subir une traversée de documents](#) en fonction de *traversable* et de *targetStep* .
5. Si *checkForUserCancelation* est vrai et que le résultat de [la vérification si le déchargement est annulé par l'utilisateur](#) étant donné *navigablesCrossingDocuments* étant donné *traversable* et *targetStep* est vrai, alors retournez.

*Certains algorithmes [vérifient si le déchargement est annulé par l'utilisateur](#) comme condition préalable à la modification de l'arborescence de l'historique. Ces algorithmes définiront *checkForUserCancelation* sur *false* lors de l'appel de cet algorithme pour éviter d'effectuer la vérification deux fois.*

Il n'est peut-être pas correct de bloquer les résultats avant le téléchargement ici. Cela peut avoir des conséquences observables.

6. Laissez `changingNavigables` être le résultat de `get all navigables` dont l'entrée d'historique de [la session actuelle changera ou rechargera](#) en fonction de *traversable* et *targetStep*.
7. Laissez *nonchangingNavigablesThatStillNeedUpdates* être le résultat de [l'obtention de tous les éléments navigables qui n'ont besoin que de la mise à jour de la longueur/de l'index de l'objet d'historique](#) en fonction de *traversable* et de *targetStep*.
8. [Pour chaque](#) *navigable* de *navigables changeants* :
  1. Soit *targetEntry* le résultat de [l'obtention de l'entrée d'historique cible](#) donnée *navigable* et *targetStep*.
  2. Définissez [l'entrée de l'historique de la session en cours](#) de *navigable* sur *targetEntry*.
  3. Définissez [la navigation en cours de](#) *navigable* sur `" ".traversal`
9. Soit *totalChangeJobs* la [taille](#) de *changingNavigables*. \_
10. Soit *terminéChangeJobs* égal à 0.
11. Soit *changingNavigableContinuations* une [file d'](#) attente vide d'états de [continuation navigables changeants](#).

*Cette file d'attente permet de scinder les opérations de modification des Navigables en deux parties. Plus précisément, `changingNavigableContinuations` contient des données pour la [deuxième partie](#).*

12. [Pour chaque](#) *navigable* de *ChangingNavigables*, [mettez en file d'attente une tâche globale](#) sur la source de [la tâche de navigation et de traversée de la fenêtre active](#) de *navigable* pour exécuter les étapes :

*Cet ensemble d'étapes est divisé en deux parties pour permettre de traiter les navigations synchrones avant le téléchargement des documents. L'état est stocké dans `changingNavigableContinuations` pour la [deuxième partie](#).*

1. Soit *displayEntry* l' [entrée d'historique de session active](#) de *navigable*.
2. Soit *targetEntry* [l'entrée de l'historique de la session en cours](#) de *navigable*.
3. Soit *changingNavigableContinuation* un état de [continuation navigable changeant](#) avec :

[document affiché](#)



document d'entrée affiché

entrée cible

entrée cible

navigable

navigable

mise à jour uniquement

FAUX

4. Si *displayEntry* est *targetEntry* et que le rechargement en attente de l'état du document de *targetEntry* est faux, alors :
  1. Définissez la mise à jour uniquement *dechangingNavigableContinuation* sur true.
  2. Mettre en *file d'attente* *dechangingNavigableContinuation* *surchangingNavigableContinuations* .
  3. Abandonnez ces étapes.

*Ce cas se produit en raison d'une navigation synchrone qui a déjà mis à jour l' entrée active de l'historique de la session .*

5. Soit *oldOrigin* l' origine de l'état du document de *targetEntry* .
6. Si le document de *targetEntry* est nul, ou si le rechargement en attente de l'état du document de *targetEntry* est vrai, alors :
  1. Laissez *navTimingType* être " *back forward*" si le document de *targetEntry* est nul ; sinon " *.reload*" .
  2. Soit *targetSnapshotParams* le résultat de l'instantané des paramètres d'instantané cible donnés *navigable* .
  3. Laissez *potentiellementTargetSpecificSourceSnapshotParams* être *sourceSnapshotParams* .
  4. Si *potentialTargetSpecificSourceSnapshotParams* est null, définissez-le sur le résultat de l'instantané des paramètres de l'instantané source en fonction du document actif de *navigable* .

*Dans ce cas, il n'y a pas de source claire de traversée/rechargement. Nous traitons cette situation comme si *navigable* naviguait lui-même, mais notez que certaines propriétés de l'initiateur d'origine de *targetEntry* sont conservées dans le document state de *targetEntry* , telles que l' initiateur origin et referrer , qui influenceront de manière appropriée la navigation.*

5. Définissez le rechargement en attente de l'état du document de *targetEntry* sur false.



6. Soit `allowPOST` le [rechargement en attente](#) de l'état du [document de](#) `targetEntry` .
7. [En parallèle](#) , [tentez de remplir le document de l'entrée d'historique](#) pour `targetEntry` , étant donné `navigable` , `potentiellementTargetSpecificSourceSnapshotParams` , `targetSnapshotParams` , avec [allowPOST](#) défini sur `allowPOST` et [CompletionSteps](#) défini pour [mettre en file d'attente une tâche globale](#) sur la [source de tâche de navigation et de traversée](#) étant donné que [la fenêtre active](#) de `navigable` s'exécute *après* `DocumentPopulated` .

Sinon, exécutez [immédiatement](#) `afterDocumentPopulated` .

Dans les deux cas, laissez `afterDocumentPopulated` être les étapes suivantes :

8. Si [le document](#) de `targetEntry` est nul, définissez la [mise à jour uniquement](#) `dechangingNavigableContinuation` sur `true`.

*Cela signifie que nous avons essayé de remplir le document, mais n'y sommes pas parvenus, par exemple parce que le serveur a renvoyé un 204.*

9. Si l' [origine](#) du [document de](#) `targetEntry` n'est pas `oldOrigin` , définissez [l'état sérialisé](#) de `targetEntry` sur [StructuredSerializeForStorage](#) (`null`).

*Cela efface l'état de l'historique lorsque l'origine a changé par rapport à un chargement précédent de `targetEntry` sans qu'une redirection ne se produise. Cela peut se produire en raison d'un changement dans les en-têtes du bac à sable CSP.*

10. Si toutes les conditions suivantes sont vraies :

- [le parent](#) de `navigable` est `null` ;
- [Le contexte de navigation](#) du [document](#) de `targetEntry` n'est pas un [contexte de navigation auxiliaire](#) dont [le contexte de navigation d'ouverture](#) n'est pas nul ; et
- [L'origine](#) du [document](#) de `targetEntry` n'est pas `oldOrigin`

puis définissez le nom de [la cible navigable](#) de l'état du [document de](#) `targetEntry` sur la chaîne vide.

11. [Mettre](#) en file d'attente `dechangingNavigableContinuation` sur `changingNavigableContinuations` .

*Le reste de ce travail [s'exécute plus tard](#) dans cet algorithme.*

13. Soit *navigablesThatMustWaitBeforeHandlingSyncNavigation* un ensemble [vide](#) .

14. Alors que *completeChangeJobs* n'est pas égal à *totalChangeJobs* :

1. Si [l'étape d'application de l'historique d'application imbriquée en cours d'exécution](#) de *traversable* est fausse, alors :

1. Tandis que [l'ensemble d'algorithmes](#) de la file d'attente de traversée de [l'historique de session](#) de *traversable* [contient](#) une ou plusieurs [étapes de navigation synchrones](#) avec une [cible navigable](#) non [contenue](#) dans *navigablesThatMustWaitBeforeHandlingSyncNavigation* :

- Soit *étapes* le premier [élément](#) de l' [ensemble d'algorithmes](#) de la file d'attente de traversée de [l'historique de session](#) *traversable* qui est [des étapes de navigation synchrones](#) avec une [cible navigable](#) non [contenue](#) dans *navigablesThatMustWaitBeforeHandlingSyncNavigation* .
- [Supprimer](#) les étapes de l'ensemble d' [algorithmes de la file d'attente de traversée de l'historique](#) des sessions de *traversable* .
- Définissez [l'étape d'application de l'historique d'application imbriquée en cours d'exécution](#) de *traversable* sur true.
- Exécutez les étapes .
- Définissez [l'étape d'historique d'application imbriquée en cours d'exécution](#) de *traversable* sur false.

*Les navigations synchrones qui sont censées avoir lieu avant cette traversée sautent la file d'attente à ce stade, afin qu'elles puissent être ajoutées au bon endroit dans les [entrées d'historique](#) de session de traversable avant que cette traversée ne décharge potentiellement leur document. [Plus de détails peuvent être trouvés ici](#) .*

2. Supposons que *changingNavigableContinuation* soit le résultat du [retrait](#) de la file d'attente *dechangingNavigableContinuations* .

3. Si la modification de *NavigableContinuation* n'est rien, [continuez](#) .

4. Laissons *displayDocument* changer [le document affiché](#) de *NavigableContinuation* .

5. Laissez *targetEntry* changer [l'entrée cible](#) de *NavigableContinuation* .

6. Soit *navigable* *changeantNavigableContinuation* 's navigable . [\\_](#)

7. Définissez [la navigation en cours](#) de *navigable* sur null.

*Cela permet à de nouvelles [navigations](#) de *navigables* de démarrer, alors que lors du parcours elles étaient bloquées.*

8. Soit ( *scriptHistoryLength* , *scriptHistoryIndex* ) le résultat de [l'obtention de la longueur et de l'index de l'objet d'historique](#) donné *traversable* et *targetStep* .

*Ces valeurs peuvent avoir changé depuis leur dernier calcul.*

9. [Ajoutez](#) *navigable* à *navigablesThatMustWaitBeforeHandlingSyncNavigation* .

*Une fois qu'un élément *navigable* a atteint ce point de traversée, des étapes de navigation synchrones supplémentaires mises en file d'attente sont susceptibles de se produire après cette traversée plutôt qu'avant, de sorte qu'elles ne sautent plus la file d'attente. [Plus de détails peuvent être trouvés ici](#) .*

10. [Mettez en file d'attente une tâche globale](#) sur la [source de la tâche de navigation et de traversée](#) en fonction de [la fenêtre active](#) de *navigable* pour exécuter les étapes :

1. Si le paramètre [update-only](#) *dechangingNavigableContinuation* est faux, alors :
  - [Décharger](#) le document affiché du document [de](#) l'entrée cible .
  - Pour chaque *childNavigable* des [navigables descendants](#) de *disabledDocument* , [mettez en file d'attente une tâche globale](#) sur la [source de la tâche de navigation et de traversée](#) en fonction de [la fenêtre active](#) de *childNavigable* pour [décharger le document actif](#) de *childNavigable* .
  - [Activez l'entrée d'historique](#) *targetEntry* pour *navigable* .
2. Si le [document](#) de *targetEntry* n'est pas égal à *disabledDocument* , mettez [en file d'attente une tâche globale](#) sur la [source de la tâche de navigation et de traversée](#) en fonction de [l'objet global pertinent](#) du [document de](#) *targetEntry* pour effectuer l'étape suivante. Sinon, continuez pour effectuer l'étape suivante dans la tâche actuellement en file d'attente.
3. [Mettre à jour le document pour l'application de l'étape d'historique](#) en fonction du [document](#) de *targetEntry* , de *targetEntry* , de [la mise à jour uniquement](#) de la *modificationNavigableContinuation* , de *scriptHistoryLength* et de *scriptHistoryIndex* .

4. Incrément *terminéChangeJobs* .

15. Soit *totalNonchangingJobs* la taille de *nonchangingNavigablesThatStillNeedUpdates* .

*Cette étape attend délibérément la fin de toutes les opérations précédentes, car elles incluent le traitement des navigations synchrones qui publieront également des tâches pour mettre à jour la longueur et l'index de l'historique.*

16. Soit *terminéNonchangingJobs* égal à 0.

17. Soit ( *scriptHistoryLength* , *scriptHistoryIndex* ) le résultat de l'obtention de la longueur et de l'index de l'objet d'historique donné *traversable* et *targetStep* .

18. Pour chaque *navigable* de *nonchangingNavigablesThatStillNeedUpdates* , mettez en file d'attente une tâche globale sur la source de la tâche de navigation et de traversée en fonction de la fenêtre active de *navigable* pour exécuter les étapes :

1. Soit *document* le document actif de *navigable* .
2. Définissez l' index de l' objet d' historique du *document* sur *scriptHistoryIndex* .
3. Définissez la longueur de l'objet d'historique du *document* sur *scriptHistoryLength* .
4. Incrément *terminéNonchangingJobs* .

19. Attendez que *completeNonchangingJobs* soit égal à *totalNonchangingJobs* .

20. Définissez l'étape d'historique de la session en cours de *traversable* sur *targetStep* .

Pour **activer l'entrée d'historique de session d'entrée d' historique** pour *navigable* *navigable* :

1. Enregistrer l'état persistant dans l' entrée d'historique de session active du *navigable* .
2. Soit *newDocument* le document de l'entrée .
3. Assert : *newDocument* is initial *about:blank* est false, c'est-à-dire que nous ne revenons jamais à l' initiale *about:blank Document* car elle est toujours remplacée lorsque nous nous en éloignons.
4. Définissez l'entrée de l'historique des sessions actives de *navigable* sur *entry* .
5. Rendre actif *newDocument* .

Pour **obtenir le pas utilisé** étant donné un [traversable navigable](#) *traversable* et un pas entier non négatif , effectuez les étapes suivantes. Ils renvoient un entier non négatif.

1. Soit *étapes* le résultat de [l'obtention de toutes les étapes d'historique utilisées](#) dans *traversable* .
2. Renvoie le plus grand [élément](#) en *étapes* inférieur ou égal à *step* .

*Cela répond aux situations où il n'y a pas [d'entrée d'historique de session](#) avec [step](#) *step* , en raison de la suppression d'un [navigable](#) .*

Pour **obtenir la longueur et l'index de l'objet d'historique** en fonction d'un [traversable navigable](#) *traversable* et d'un entier non négatif *step* , procédez comme suit. Ils renvoient un [tuple](#) de deux entiers non négatifs.

1. Soit *étapes* le résultat de [l'obtention de toutes les étapes d'historique utilisées](#) dans *traversable* .
2. Soit *scriptHistoryLength* la [taille](#) des *étapes* .
3. [Assert](#) : *étapes* [contient](#) *étape* .

*On suppose que le pas a été ajusté en [récupérant le pas utilisé](#) .*

4. Soit *scriptHistoryIndex* l'index de *step* in *steps* .
5. Retour ( *scriptHistoryLength* , *scriptHistoryIndex* ).

Pour **obtenir tous les éléments navigables dont l'entrée d'historique de la session en cours sera modifiée ou rechargée** en fonction d'un [traversable navigable traversable](#) et d'un entier non négatif *targetStep* , procédez comme suit. Ils renvoient une [liste](#) de [navigables](#) .

1. Soit *les résultats* une [liste](#) vide .
2. Soit *navigablesToCheck* « *traversable* ».

*Cette liste est étendue dans la boucle ci-dessous.*

3. [Pour chaque](#) *navigable* de *navigablesToCheck* :
  1. Soit *targetEntry* le résultat de [l'obtention de l'entrée d'historique cible](#) donnée *navigable* et *targetStep* .
  2. Si *targetEntry* n'est pas [l'entrée d'historique](#) de la session en cours de *navigable* ou si [le rechargement en attente](#) de l'état du [document](#) de *targetEntry* est vrai, alors [ajoutez](#) *navigable* aux *résultats* .
  3. Si le [document de](#) *targetEntry* est [le document](#) de *navigable* et que [le rechargement en attente](#) de l'état du [document](#) de *targetEntry* est faux,

alors étendez *navigablesToCheck* avec les navigables enfants de *navigable* .

*L'ajout d'éléments navigables enfants à *navigablesToCheck* signifie que ces éléments navigables seront également vérifiés par cette boucle. Les objets navigables enfants ne sont vérifiés que si le document actif de l'objet navigable ne change pas dans le cadre de ce parcours.*

4. Renvoyer les résultats .

Pour **obtenir tous les éléments navigables qui nécessitent uniquement une mise à jour de la longueur/de l'index de l'objet d'historique** en fonction d'un traversable navigable traversable et d'un entier non négatif *targetStep* , procédez comme suit. Ils renvoient une liste de navigables .

*Les autres voies navigables pourraient ne pas être impactées par la traversée. Par exemple, si la réponse est un 204, le document actuellement actif restera. De plus, revenir en arrière après un 204 modifiera l'entrée actuelle de l'historique de la session , mais l'entrée active de l'historique de la session sera déjà correcte.*

1. Soit les résultats une liste vide .
2. Soit *navigablesToCheck* « traversable ».

*Cette liste est étendue dans la boucle ci-dessous.*

3. Pour chaque *navigable* de *navigablesToCheck* :
  1. Soit *targetEntry* le résultat de l'obtention de l'entrée d'historique cible donnée *navigable* et *targetStep* .
  2. Si *targetEntry* est *navigable* , l'entrée d'historique de la session actuelle de *targetEntry* et le rechargement en attente de l'état du document de *targetEntry* est faux, alors :
    1. Ajouter *navigable* aux résultats .
    2. Étendez *navigablesToCheck* avec les navigables enfants de *navigable* .

*L'ajout d'éléments navigables enfants à *navigablesToCheck* signifie que ces éléments navigables seront également vérifiés par cette boucle. les objets navigables enfants ne sont vérifiés que si le document actif de l'objet navigable ne changera pas dans le cadre de ce parcours.*

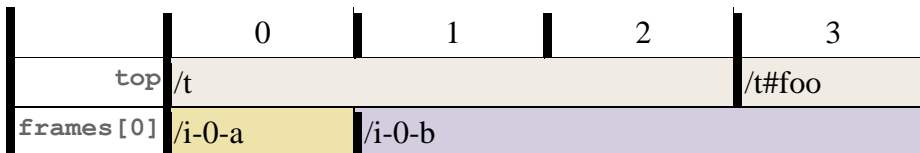
4. Renvoyer les résultats .

Pour **obtenir l'entrée d'historique cible** en fonction d'un *navigable* navigable et d'un entier non négatif *step* , procédez comme suit. Ils renvoient une entrée d'historique de session .

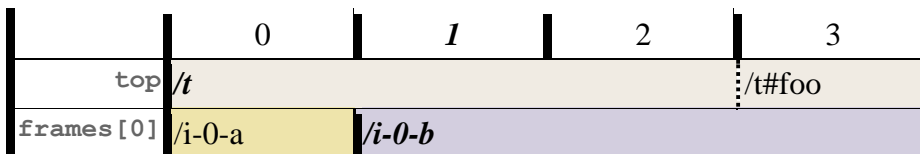


1. Laissez les entrées résulter de [l'obtention des entrées de l'historique de session](#) pour *navigable* .
2. Renvoie l' [élément](#) dans les entrées dont le plus grand [pas](#) est inférieur ou égal à *step* .

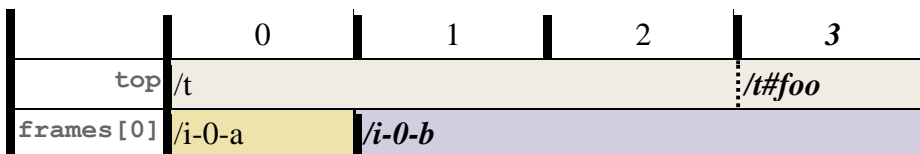
Pour voir pourquoi [l'obtention de l'entrée d'historique cible](#) renvoie l'entrée avec le plus grand [pas](#) inférieur ou égal au pas d'entrée, considérez le [diagramme de Jake](#) suivant :



Pour l'étape d'entrée 1, l'entrée d'historique cible pour le `top` *navigable* est l' `/t` entrée dont le [pas](#) est 0, tandis que l'entrée d'historique cible pour le `frames[0]` *navigable* est l' `/i-0-b` entrée dont le [pas](#) est 1 :



De même, étant donné le pas d'entrée 3, nous obtenons l' `top` entrée dont le [pas](#) est 3 et l' `frames[0]` entrée dont le [pas](#) est 1 :



Pour **obtenir tous les éléments navigables susceptibles de subir une traversée de documents** à partir d'un traversable [navigable traversable](#) et d'un entier non négatif *targetStep* , procédez comme suit. Ils renvoient une [liste](#) de [navigables](#) .

*Du point de vue de la [file d'attente de traversée de l'historique de session](#) de traversable , ces documents sont candidats pour passer d'un document à l'autre pendant la traversée décrite par *targetStep* . Ils ne subiront pas de parcours inter-documents si le code d'état de leur document cible est HTTP 204 No Content.*

*Notez que si un [navigable](#) donné peut subir une traversée inter-documents, cet algorithme renverra [navigable](#) mais pas ses [enfants navigables](#) . Ceux-ci se retrouveraient [déchargés](#) , non parcourus.*

1. Soit les résultats une [liste](#) vide .
2. Soit *navigablesToCheck* « traversable ».

*Cette liste est étendue dans la boucle ci-dessous.*

3. Pour chaque *navigable* de *navigablesToCheck* :

1. Soit *targetEntry* le résultat de l'obtention de l'entrée d'historique cible donnée *navigable* et *targetStep* .
2. Si le document de *targetEntry* n'est pas le document de *navigable* ou si le rechargement en attente de l'état du document de *targetEntry* est vrai, alors ajoutez *navigable* aux *résultats* .

*Bien que l'entrée d'historique active de *navigable* puisse changer de façon synchrone, la nouvelle entrée aura toujours la même valeur , donc l'accès au document de *navigable* est fiable.*Document

3. Sinon, étendez *navigablesToCheck* avec les navigables enfants de *navigable* .

*L'ajout d'éléments navigables enfants à *navigablesToCheck* signifie que ces éléments navigables seront également vérifiés par cette boucle. Les objets navigables enfants ne sont vérifiés que si le document actif de l'objet *navigable* ne change pas dans le cadre de ce parcours.*

4. Renvoyer les *résultats* .

#### 7.4.6.2 Mise à jour du document

Pour **mettre à jour le document pour l'application de l'étape d'historique** étant donné un Document *document* , une entrée d'historique de session , un booléen *doNotReactivate* et des entiers *scriptHistoryLength* et *scriptHistoryIndex* :

1. Laissez *documentIsNew* être vrai si la dernière entrée du *document* est nulle ; sinon faux.
2. Laissez *documentsEntryChanged* être vrai si la dernière entrée du *document* n'est pas *entry* ; sinon faux.
3. Définissez l'index de l' objet d'historique du *document* sur *scriptHistoryIndex* .
4. Définissez la longueur de l'objet d'historique du *document* sur *scriptHistoryLength* .
5. Si *documentsEntryChanged* est vrai, alors :
  1. Soit *oldURL* l' URL de la dernière entrée du *document* .
  2. Définir la dernière entrée du *document* sur *entrée* .



3. [Restaure l'état de l'objet d'historique](#) en fonction du *document* et de l'entrée .
4. Si *documentIsNew* a la valeur false, [déclenchez un événement](#) nommé `popstate` au niveau de l'[objet global pertinent du document](#) , en utilisant , avec l'attribut initialisé à l' [état de l'objet historique](#) du *document* .`PopStateEventstate`
5. [Restaurer l'état persistant](#) de l'entrée .
6. Si *documentIsNew* est faux et que [le fragment](#) de *oldURL* n'est pas égal au [fragment](#) de l'[URL](#) de l'entrée , [placez une tâche globale en file d'attente](#) sur la [source de la tâche de manipulation DOM](#) en fonction [de l'objet global pertinent](#) du *document* pour [déclencher un événement](#) nommé à l' [adresse pertinente](#) du *document* . [objet global](#) , utilisant , avec l'attribut initialisé à la [sérialisation](#) de *oldURL* et l'attribut initialisé à la [sérialisation](#)`hashchangeHashChangeEventoldURLnewURL` de l'[URL](#) de l'entrée .
6. Si *documentIsNew* est vrai, alors :
  1. [Essayez de faire défiler jusqu'au fragment](#) pour *document* .
  2. À ce stade, des scripts peuvent s'exécuter pour le *document* **document nouvellement créé** .
7. Sinon, si *documentsEntryChanged* est faux et *doNotReactivate* est faux, alors [réactivez](#) *document* .

*documentsEntryChanged peut avoir la valeur false pour l'une des deux raisons suivantes : soit nous restaurons à partir de [bfcache](#) , soit nous terminons de manière asynchrone une navigation synchrone qui a déjà défini de manière synchrone [la dernière entrée](#) du document . L'argument *doNotReactivate* fait la distinction entre ces deux cas.*

Pour **restaurer l'état de l'objet d'historique** en fonction de l'entrée d'entrée d'historique `Document` de *document* et [de session](#) :

1. Soit *targetRealm* le [domaine pertinent](#) du *document* .
2. Soit *state* être [StructuredDeserialize](#) ( [état sérialisé](#) de l'entrée , *targetRealm* ). Si cela lève une exception, attrapez-la et laissez l'état être nul.
3. Définit l' [état](#) de l' [objet historique](#) du *document* sur *state* .

Pour **rendre actif** un `Document` *document* :

1. Soit *window* l' [objet global pertinent](#) du *document* .

2. Définit la valeur de l' emplacement interne de [\[\[Window\]\]](#) du [contexte de navigation](#) du *document* sur *window* .[WindowProxy](#)
3. Définissez [l'état de visibilité](#) du *document* sur [l'état de visibilité](#) du système de [l'élément navigable traversable](#) du nœud du *document* .
4. Définissez [l' indicateur prêt à l' exécution](#) de l ' [objet de paramètres pertinent](#) de *la fenêtre* .

Pour **réactiver** un [Document](#) *document* :

*Cet algorithme met à jour le document après qu'il soit sorti de [bfcache](#) , c'est-à-dire après qu'il a été rendu [complètement actif](#) à nouveau.*

1. [Pour chaque](#) *formControl* des contrôles de formulaire dans *le document* avec un [nom de champ de remplissage automatique](#) de " [off](#) ", appelez l' [algorithme de réinitialisation](#) pour *formControl* .
2. Si [les poignées du minuteur suspendu](#) du *document* ne sont pas [vides](#) :
  1. [Assert](#) : [le temps de suspension](#) du *document* n'est pas nul.
  2. Soit *suspendDuration* le [temps de haute résolution actuel](#) moins [le temps de suspension](#) du *document* .
  3. Soit *activeTimers* la carte des [temporisateurs actifs de l'objet global pertinent](#) du *document* .
  4. Pour chaque *poignée* dans [les poignées de minuterie suspendues](#) du *document* , si *activeTimers* [ *poignée* ] [existe](#) , alors augmentez *activeTimers* [ *poignée* ] de *suspendDuration* .
3. Si [la préparation actuelle](#) du *document* du *document* est " " et que la page du *document* [affichant](#) l'indicateur est fausse, alors :[complete](#)
  1. Définissez la page du *document* [affichant](#) l'indicateur sur vrai.
  2. [Mettez à jour l'état de visibilité](#) du *document* sur " [visible](#) ".
  3. [Déclenchez un événement de transition de page](#) nommé [pageshow](#) au niveau [de l'objet global pertinent](#) du *document* avec *true*.

Pour **essayer de faire défiler jusqu'au fragment** d'un [Document](#) *document* , effectuez les étapes suivantes [en parallèle](#) :

1. Attendez un laps de temps [défini par l'implémentation](#) . (Ceci est destiné à permettre à l'agent utilisateur d'optimiser l'expérience utilisateur face aux problèmes de performances.)

2. Mettez en file d'attente une tâche globale sur la source de la tâche de navigation et de parcours en fonction de l'objet global pertinent du *document* pour exécuter ces étapes :
  1. Si le *document* n'a pas d'analyseur, ou si son analyseur a arrêté l'analyse , ou si l'agent utilisateur a des raisons de croire que l'utilisateur n'est plus intéressé à faire défiler jusqu'au fragment , alors abandonnez ces étapes.
  2. Faites défiler jusqu'au fragment de *document* donné .
  3. Si la partie indiquée de *document* est toujours nulle, essayez de faire défiler jusqu'au fragment pour *document* .

#### 7.4.6.3 Défilement vers un fragment

Pour **défiler jusqu'au fragment** donné à un Document *document* :

1. Si la partie indiquée du *document* est null, définissez l'élément cible du *document* sur null.
2. Sinon, si la partie indiquée du *document* est en haut du document , alors :
  1. Définissez l'élément cible du *document* sur null.
  2. Faites défiler jusqu'au début du document pour *document* . [VUE CSSOM]
  3. Retour.
3. Sinon:
  1. Assert : la partie indiquée du *document* est un élément.
  2. Soit *target* la partie indiquée du *document* .
  3. Définissez l'élément cible du *document* sur *target* .
  4. Exécutez l' algorithme révélant les détails de l'ancêtre sur *la cible* .
  5. Exécutez l' algorithme de révélation de l'ancêtre caché jusqu'à ce qu'il soit trouvé sur *la cible* .
  6. Faites défiler la cible dans la vue , avec le comportement défini sur "auto", le *bloc* défini sur "démarrer" et l'*inline* défini sur "le plus proche". [VUE CSSOM]
  7. Exécutez les étapes de mise au point pour *la cible* , avec la fenêtreDocument de la comme *cible de repli* .

8. Déplacez le [point de départ de la navigation de mise au point séquentielle](#) vers *la cible* .

La **partie indiquée**[Document](#) d'un est celle que son [fragment](#) d' [URL](#) identifie, ou nul si le fragment n'identifie rien. La sémantique du [fragment](#) en termes de mappage vers un nœud est définie par la spécification qui définit le [type MIME](#) utilisé par le (par exemple, le traitement des [fragments](#) pour [les types XML MIME](#) est de la responsabilité de RFC7303). [\[RFC7303\]](#)[Document](#)

Il existe également un **élément cible** pour chaque [Document](#), qui est utilisé pour définir la `:target` pseudo-classe et est mis à jour par l'algorithme ci-dessus. Il est initialement nul.

Pour un *document* [document HTML](#) , le modèle de traitement suivant doit être suivi pour déterminer sa [partie indiquée](#) :

1. Soit *fragment* le [fragment](#) de l' [URL](#) du *document* .
2. Si *fragment* est la chaîne vide, renvoie la valeur spéciale **top of the document** .
3. Soit *potentialIndicatedElement* le résultat de [la recherche d'un élément indiqué potentiel](#) donné *document* et *fragment* .
4. Si *potentialIndicatedElement* n'est pas nul, alors retourne *potentialIndicatedElement* .
5. Soit *fragmentBytes* le résultat du [pourcentage de décodage](#) *fragment* .
6. Soit *decodedFragment* le résultat de l'exécution [du décodage UTF-8 sans BOM](#) sur *fragmentBytes* .
7. Définissez *potentialIndicatedElement* sur le résultat de [la recherche d'un élément indiqué potentiel](#) donné *document* et *decodedFragment* .
8. Si *potentialIndicatedElement* n'est pas nul, alors retourne *potentialIndicatedElement* .
9. Si *decodedFragment* est une correspondance [ASCII non sensible à la casse](#) pour la chaîne `top`, renvoie alors le [haut du document](#) .
10. Renvoie nul.

Pour **trouver un élément indiqué potentiel** étant donné un [Document](#) *document* et un *fragment* de chaîne , exécutez ces étapes :

1. S'il existe un élément [dans l'arborescence du document](#) dont [la racine](#) est *document* et qui a un [ID](#) égal à *fragment* , alors renvoie le premier élément de ce type dans [l'ordre de l'arborescence](#) .

2. S'il y a un aélément dans l'arborescence du document dont la racine est *document* qui a un nameattribut dont la valeur est égale à *fragment* , alors retournez le premier élément de ce type dans l'ordre de l'arborescence .
3. Renvoi nul.

#### 7.4.6.4 État d'entrée d'historique persistant

Pour **enregistrer l'état persistant** dans une *entrée* d'historique de session :

1. Définissez les données de position de défilement de *l'entrée* pour qu'elles contiennent les positions de défilement de toutes les régions de défilement restaurables du document de *l'entrée* .
2. Facultativement, mettez à jour l'état utilisateur persistant de *l'entrée* pour refléter tout état que l'agent utilisateur souhaite conserver, comme les valeurs des champs de formulaire.

Pour **restaurer l'état persistant** à partir d'une *entrée* d'historique de session :

1. Si le mode de restauration du défilement de *l'entrée* est " " , alors restaurer les données de position de défilement données à *l'entrée* .auto

*Le fait que l'agent utilisateur ne restaure pas les positions de défilement n'implique pas que les positions de défilement seront laissées à une valeur particulière (par exemple, (0,0)). La position de défilement réelle dépend du type de navigation et de la stratégie de mise en cache particulière de l'agent utilisateur. Ainsi, les applications Web ne peuvent pas assumer une position de défilement particulière, mais sont plutôt invitées à la définir comme elles le souhaitent.*

2. Facultativement, mettez à jour d'autres aspects du document de *l'entrée* et de son rendu, par exemple les valeurs des champs de formulaire, que l'agent utilisateur avait précédemment enregistrées dans l'état utilisateur persistant de *l'entrée* .

*Cela peut même inclure la mise à jour de l' dirattribut des textareaéléments ou inputdes éléments dont typel'attribut est à l' état Texte ou à l' état Recherche , si l'état persistant inclut la directionnalité de l'entrée utilisateur dans ces contrôles.*

*La restauration de la valeur des contrôles de formulaire dans le cadre de ce processus ne déclenche aucun événement input ou , mais peut déclencher les éléments personnalisés associés au formulaire .change`formStateRestoreCallback`*

---

Chacun Document a un booléen **qui a été défilé par l'utilisateur** , initialement faux. Si l'utilisateur fait défiler le document, l'agent utilisateur doit définir ce document a été fait défiler par l'utilisateur sur vrai.

Les **régions de défilement restaurables** d'un Document document sont la fenêtre d'affichage du document et toutes les régions de défilement du document à l'exception des conteneurs navigables .

*La restauration de défilement de navigation enfant est gérée dans le cadre de la restauration d'état pour l' entrée d'historique de session pour ces éléments navigables Document .*

Pour **restaurer les données de position de défilement** en fonction d'une entrée d'historique de session :

1. Soit document le document d' entrée .
2. Si document 's has been scrolled by the user is true, alors l'agent utilisateur doit revenir.
3. L'agent utilisateur devrait essayer d'utiliser les données de position de défilement de l'entrée pour restaurer les positions de défilement des régions de défilement restaurables du document de l'entrée . L'agent utilisateur peut continuer à essayer de le faire périodiquement, jusqu'à ce que le document 's a été fait défiler par l'utilisateur devienne vrai.

*Ceci est formulé comme une tentative , qui est potentiellement répétée jusqu'au succès ou jusqu'à ce que l'utilisateur fasse défiler, en raison du fait que le contenu pertinent indiqué par les données de position de défilement peut prendre un certain temps à se charger à partir du réseau. La restauration du défilement peut être affectée par l'ancrage du défilement. [CSSSCROLLANCHORING]*

1. [7.5 Cycle de vie des documents](#)
  1. [7.5.1 Infrastructure de création de documents partagés](#)
  2. [7.5.2 Chargement de documents HTML](#)
  3. [7.5.3 Chargement de documents XML](#)
  4. [7.5.4 Chargement de documents texte](#)
  5. [7.5.5 Chargement multipart/x-mixed-replace de documents](#)
  6. [7.5.6 Chargement de documents multimédias](#)
  7. [7.5.7 Chargement d'un document pour un contenu en ligne qui n'a pas de DOM](#)
  8. [7.5.8 Terminer le processus de chargement](#)
  9. [7.5.9 Déchargement des documents](#)
  10. [7.5.10 Destruction de documents](#)
  11. [7.5.11 Abandon du chargement d'un document](#)
2. [7.6 L'X-Frame-Options en-tête ``](#)
3. [7.7 L'Refresh en-tête ``](#)
4. [7.8 Considérations sur l'interface utilisateur du navigateur](#)

## 7.5 Cycle de vie des documents

### 7.5.1 Infrastructure de création de documents partagés

Lors du chargement d'un document à l'aide de l'un des algorithmes ci-dessous, nous utilisons les étapes suivantes pour **créer et initialiser un Document objet**, étant donné un [type](#) `type`, [un type de contenu](#) `contentType` et [des paramètres de navigation](#) `navigationParams` :

`Document` des objets sont également créés lors [de la création d'un nouveau contexte de navigation et d'un nouveau document](#) ; ces [initiales](#) `about:blank Document` ne sont jamais créées par cet algorithme. De plus, des objets sans [contexte de navigation](#) `Document` peuvent être créés via diverses API, telles que `document.implementation.createHTMLDocument()`.

1. Soit `browserContext` le [contexte de navigation actif](#) du paramètre [navigable](#) de `navigationParams`.
2. Définissez `browserContext` sur le résultat de l' [obtention d'un contexte de navigation à utiliser pour une réponse de navigation](#) en fonction de `scanningContext`, de l'ensemble final d' [indicateurs de sandboxing de navigationParams](#), [de la stratégie d'ouverture croisée](#) de `navigationParams` et [du résultat d'application COOP](#) de `navigationParams`.



Cela peut entraîner un changement de groupe de contexte de navigation , auquel cas `browserContext` sera un contexte de navigation nouvellement créé au lieu d'être le contexte de navigation actif de `navigationParams` . Dans ce cas, les agents , , et créés ne seront finalement pas utilisés ; parce que l' origine du créé est opaque , nous finirons par créer un nouvel agent et plus tard dans cet algorithme pour accompagner le nouveau `.WindowDocumentDocumentWindow Document`

3. Soit `permissionsPolicy` le résultat de la création d'une politique d'autorisations à partir d'une réponse donnée du conteneur navigable de `navigationParams` , de l'origine de `navigationParams` et de la réponse de `navigationParams` . [POLITIQUE D'AUTORISATION]

La création d'une stratégie d'autorisations à partir d'un algorithme de réponse utilise l' origine transmise . Si `document.domain` a été utilisé pour le document conteneur navigable de `navigationParams` , son origine ne peut pas être le même domaine d'origine que l'origine transmise, car ces étapes s'exécutent avant la création du document , il ne peut donc pas encore avoir utilisé . Notez que cela signifie que les vérifications de la politique d'autorisations sont moins permissives par rapport à la vérification de la même origine à la place `document.domain`

Voir ci-dessous pour quelques exemples de ceci en action.

4. Soit `creationURL` l' URL de la réponse de `navigationParams` .
5. Si la demande de `navigationParams` n'est pas nulle, définissez `creationURL` sur l' URL actuelle de la demande de `navigationParams` .
6. Soit `window` nul.
7. Si le document actif de `scanningContext` est initial est vrai et que l'origine du document actif de `scanningContext` est le même domaine d'origine que l'origine de `navigationParams` , alors définissez `window` sur la fenêtre active de `scanningContext` `.about:blank`

Cela signifie que le initial `about:blank Document` et le nouveau `Document` qui est sur le point d'être créé partageront le même `Window` objet.

8. Sinon:
  1. Soit `oacHeader` le résultat de l'obtention d'une valeur de champ structuré donnée par ``Origin-Agent-Cluster`` et `" item"` à partir de la liste d'en-têtes de la réponse de `navigationParams` .
  2. Soit `requestsOAC` vrai si `oacHeader` n'est pas nul et `oacHeader [0]` est le booléen vrai ; sinon faux.
  3. Si l'environnement réservé de `navigationParams` est un contexte non sécurisé , définissez `requestsOAC` sur `false`.



4. Soit *agent* le résultat de [l'obtention d'un agent de fenêtre d'origine similaire](#) en fonction de [l'origine](#) de *navigationParams* , du [groupe](#) de *scanningContext* et de *requestsOAC* .
5. Soit *realmExecutionContext* le résultat de [la création d'un nouvel agent](#) donné par le domaine et des personnalisations suivantes :
  - Pour l'objet global, créez un nouvel [Window](#) objet.
  - Pour la liaison globale **this** , utilisez l'objet de *scanningContext* [WindowProxy](#) .
6. Définissez *window* sur l' [objet global](#) du composant Realm de *realmExecutionContext* .
7. Soit *topLevelCreationURL* être *creationURL* .
8. Soit *topLevelOrigin* l' [origine](#) de *navigationParams* .
9. Si [le conteneur](#) de *navigable* n'est pas nul, alors :
  - Soit *parentEnvironment* l' objet de [paramètres pertinent](#) du [conteneur](#) *navigable* .
  - Définissez *topLevelCreationURL* sur l' [URL de création de niveau supérieur](#) de *parentEnvironment* .
  - Définissez *topLevelOrigin* sur l'[origine de niveau supérieur](#) de *parentEnvironment* .
10. [Configurez un objet de paramètres d'environnement de fenêtre](#) avec *creationURL* , *realmExecutionContext* , l'[environnement réservé](#) de *navigationParams* , *topLevelCreationURL* et *topLevelOrigin* .

*C'est le cas habituel, où le nouveau [Document](#) que nous sommes sur le point de créer est [Window](#) accompagné d'un nouveau.*

9. Soit *loadTimingInfo* une nouvelle [information de synchronisation de chargement de document](#) avec son [heure de début de navigation](#) définie sur l'[heure de début](#) de l'[information de synchronisation](#) de la [réponse](#) de *navigationParams* .
10. Soit *document* un new [Document](#), avec

[taper](#)

*taper*

[type de contenu](#)

*type de contenu*

[origine](#)

[origine](#) de *navigationParams*

[contexte de navigation](#)

*contexte de navigation*

**conteneur de stratégie**

Conteneur de stratégie de *navigationParams*

**politique d'autorisations**

*permissionsPolicy*

**ensemble d'indicateurs de sandboxing actif**

Ensemble final d'indicateurs de sandboxing de *navigationParams*

**politique d'ouverture d'origine croisée**

Politique d'ouverture cross-origin de *navigationParams*

**informations de synchronisation de chargement**

*loadTimingInfo*

**a été créé via des redirections cross-origin**

La réponse de *navigationParams* contient des redirections d'origine croisée

**identifiant de navigation**

ID de *navigationParams*

**URL**

*créationURL*

**état de préparation actuel des documents**

"loading"

11. Définir la *fenêtre associée* *Document* au *document* .
12. Exécutez l'initialisation CSP pour un *Document* *document* donné . [CSP]
13. Si la requête de *navigationParams* n'est pas nulle, alors :
  1. Définissez le référent du *document* sur la chaîne vide.
  2. Soit *referrer* le référent de la requête de *navigationParams* .
  3. Si le *référent* est un enregistrement d'URL , définissez le référent du *document* sur la sérialisation du *référent* .

*Par Fetch , le référent sera soit un enregistrement d'URL soit "no-referrer" à ce stade.*
14. Si le contrôleur de récupération de *navigationParams* n'est pas nul, alors :
  1. Soit *fullTimingInfo* le résultat de l'extraction de toutes les informations temporelles du contrôleur d'extraction de *navigationParams* .
  2. Soit *redirectCount* égal à 0 si la réponse de *navigationParams* a des redirections d'origine croisée est true ; sinon le compte de redirection de la requête de *navigationParams* .
  3. Créez l'entrée de minutage de navigation pour le *document* , en fonction de *fullTimingInfo* , *redirectCount* , *navigationTimingType* , des informations de minutage du service worker de la réponse de *navigationParams* et des informations de corps de la réponse de *navigationParams* .
15. Créez l'entrée de minutage de navigation pour le *document* , avec les informations de minutage de la

[réponse](#) de *navigationParams* , *redirectCount* , [le type de minutage de navigation](#) de *navigationParams* et les informations de minutage du [service worker](#) de [la réponse](#) de *navigationParams* .

16. Si [la réponse](#) de *navigationParams* a un en-tête ` ` , alors [:Refresh](#)

1. Soit *value* le [décodage isomorphe](#) de la valeur de l'en-tête.
2. Exécutez les [étapes d'actualisation déclarative partagée](#) avec *document* et *value* .

Nous n'avons pas actuellement de spécification sur la façon de gérer plusieurs [Refresh](#) en-têtes ` ` . Ceci est suivi en tant que [problème #2900](#) .

17. Si [les indications précoces de validation](#) de *navigationParams* ne sont pas nulles, appelez [les indications précoces de validation](#) de *navigationParams* avec *document* .

18. [Traite les en-têtes de lien](#) donnés au *document* , [la réponse](#) de *navigationParams* et " ".pre-media

19. *Document* de retour .

Dans cet exemple, le document enfant n'est pas autorisé à utiliser [PaymentRequest](#), bien qu'il soit [le même domaine d'origine](#) au moment où le document enfant essaie de l'utiliser. Au moment où le document enfant est initialisé, seul le document parent a défini [document.domain](#), et pas le document enfant.

```
<!-- https://foo.example.com/a.html -->
<!doctype html>
<script>
document.domain = 'example.com';
</script>
<iframe src=b.html></iframe>
<!-- https://bar.example.com/b.html -->
<!doctype html>
<script>
document.domain = 'example.com'; // This happens after the document
is initialized
new PaymentRequest(...); // Not allowed to use
</script>
```

Dans cet exemple, le document enfant est autorisé à utiliser [PaymentRequest](#), bien qu'il ne soit pas [le même domaine d'origine](#) au moment où le document enfant essaie de l'utiliser. Au moment où le document enfant est initialisé, aucun des documents n'a [document.domain](#) encore été défini, donc [le même domaine d'origine](#) revient à une vérification normale [de la même origine](#) .

```

<!-- https://example.com/a.html -->
<!doctype html>
<iframe src=b.html></iframe>
<!-- The child document is now initialized, before the script below
is run. -->
<script>
document.domain = 'example.com';
</script>
<!-- https://example.com/b.html -->
<!doctype html>
<script>
new PaymentRequest(...); // Allowed to use
</script>

```

## 7.5.2 Chargement de documents HTML

Pour **charger un document HTML** , étant donné [les paramètres de navigation](#) `navigationParams` :

1. Soit *document* le résultat de [la création et de l'initialisation d'un Document objet](#) donné " html", " text/html" et *navigationParams* .
2. Créez un [analyseur HTML](#) et associez - le au *document* . Chaque [tâche](#) que la [source de la tâche réseau](#) place dans la [file d'attente des tâches](#) lors de l'extraction doit ensuite remplir le [flux d'octets d'entrée](#) de l'analyseur avec les octets extraits et obliger l' [analyseur HTML](#) à effectuer le traitement approprié du flux d'entrée.

La première [tâche](#) que la [source de la tâche réseau](#) place dans la [file d'attente des tâches](#) lors de l'extraction doit [traiter les en-têtes de lien](#) donnés *document* , [la réponse](#) de *navigationParams* et " " , après que la tâche a été traitée par l' [analyseur HTML](#) .media

Avant toute exécution de script, l'agent utilisateur doit attendre que [les scripts puissent s'exécuter pour que le document nouvellement créé](#) soit vrai pour *document* .

*Le [flux d'octets d'entrée](#) convertit les octets en caractères à utiliser dans le [tokenizer](#) . Ce processus repose, en partie, sur les informations de codage de caractères trouvées dans les véritables [métadonnées Content-Type](#) de la ressource ; le type calculé n'est pas utilisé à cette fin.*

Lorsqu'il n'y a plus d'octets disponibles, l'agent utilisateur doit [mettre en file d'attente une tâche globale](#) sur la [source de la tâche réseau](#) en fonction de l'[objet global pertinent](#) du *document* pour que l'analyseur traite le caractère

EOF implicite, ce qui provoque finalement le déclenchement d'un événement.[load](#)

### 3. Document de retour .

## 7.5.3 Chargement de documents XML

Lorsqu'ils sont confrontés à l'affichage d'un fichier XML en ligne, à condition que [les paramètres de navigation](#) *navigationParams* et un *type* de chaîne, les agents utilisateurs doivent suivre les exigences définies dans *XML* et *Namespaces in XML*, *XML Media Types*, *DOM* et d'autres spécifications pertinentes pour [créer et initialiser un Document objet](#) *document*, étant donné "*xml*", *type* et *navigationParams*, et renvoyez cela [Document](#). Ils doivent également créer un [analyseur XML](#) correspondant. [\[XML\]](#) [\[XMLNS\]](#) [\[RFC7303\]](#) [\[DOM\]](#)

*Au moment de la rédaction de cet article, la communauté des spécifications XML n'avait pas encore précisé comment XML et le DOM interagissent.*

La première [tâche](#) que la [source de la tâche réseau](#) place dans la [file d'attente des tâches](#) lors de l'extraction doit [traiter les en-têtes de lien](#) donnés *document*, la [réponse](#) de *navigationParams* et "", après que la tâche a été traitée par l' [analyseur XML](#).*media*

Les en-têtes HTTP réels et autres métadonnées, et non les en-têtes mutés ou implicites par les algorithmes donnés dans cette spécification, sont ceux qui doivent être utilisés lors de la détermination de l'encodage des caractères selon les règles données dans les spécifications ci-dessus. Une fois le codage de caractères établi, le [codage de caractères du document](#) doit être défini sur ce codage de caractères.

Avant toute exécution de script, l'agent utilisateur doit attendre que [les scripts puissent s'exécuter pour que le document nouvellement créé](#) soit vrai pour le nouveau fichier [Document](#).

Une fois l'analyse syntaxique terminée, l'agent utilisateur doit définir [l'identifiant de navigation](#) du *document* sur null.

*Pour les documents HTML, ceci est réinitialisé lorsque l'analyse est terminée, après le déclenchement de l'événement load.*

Les messages d'erreur du processus d'analyse (par exemple, les erreurs de bonne formation de l'espace de noms XML) peuvent être signalés en ligne en mutant le [Document](#).

## 7.5.4 Chargement de documents texte

Pour **charger un document texte** , à partir d'un [paramètre de navigation](#) *navigationParams* et d'un *type* de chaîne :

1. Soit *document* le résultat de [la création et de l'initialisation d'un Document objet](#) donné " *html*", *type* et *navigationParams* .
2. L'analyseur de Set *document* [ne peut pas modifier l'indicateur de mode](#) sur *true*.
3. Réglez [le mode](#) du *document* sur " *.no-quirks*
4. Créez un [analyseur HTML](#) et associez - le au *document* . Faites comme si le tokenizer avait émis un jeton de balise de début avec le nom de tag "pre" suivi d'un seul caractère U+000A LINE FEED (LF) et basculez le tokenizer de l' [analyseur HTML vers l' état PLAINTEXT](#) . Chaque [tâche](#) que la [source de la tâche réseau](#) place dans la [file d'attente des tâches](#) lors de l'extraction doit ensuite remplir le [flux d'octets d'entrée](#) de l'analyseur avec les octets extraits et obliger l' [analyseur HTML](#) à effectuer le traitement approprié du flux d'entrée.

[L'encodage](#) du *document* doit être défini sur l'encodage de caractères utilisé pour décoder le document lors de l'analyse.

La première [tâche](#) que la [source de la tâche réseau](#) place dans la [file d'attente des tâches](#) lors de l'extraction doit [traiter les en-têtes de lien](#) donnés *document* , [la réponse](#) de *navigationParams* et " " , après que la tâche a été traitée par l' [analyseur HTML](#) .*media*

Avant toute exécution de script, l'agent utilisateur doit attendre que [les scripts puissent s'exécuter pour que le document nouvellement créé](#) soit vrai pour *document* .

Lorsqu'il n'y a plus d'octets disponibles, l'agent utilisateur doit [mettre en file d'attente une tâche globale](#) sur la [source de la tâche réseau](#) en fonction de [l'objet global pertinent](#) du *document* pour que l'analyseur traite le caractère EOF implicite, ce qui provoque finalement le déclenchement d'un événement.[load](#)

5. Les agents utilisateurs peuvent ajouter du contenu à l' [head](#)élément de *document* , par exemple, en créant un lien vers une feuille de style, en fournissant un script ou en donnant au document un [title](#).

*En particulier, si l'agent utilisateur prend en charge la `Format=Flowed`fonctionnalité de la RFC 3676, l'agent utilisateur devra alors appliquer un style supplémentaire pour que le texte s'enroule correctement et pour gérer la fonctionnalité de citation. Cela pourrait être effectué en utilisant, par exemple, une extension CSS.*

6. *Document* de retour .

Les règles de conversion des octets du document en texte brut en caractères réels et les règles de rendu réel du texte à l'utilisateur sont définies par les spécifications du type [MIME calculé](#) de la ressource (c'est-à-dire *type*).

### 7.5.5 Chargement [multipart/x-mixed-replace](#) de documents

Pour **charger un [multipart/x-mixed-replace](#) document**, [les paramètres de navigation](#) donnés *navigationParams*, [les paramètres d'instantané source](#) *sourceSnapshotParams* et [l'origine](#) *initiatorOrigin* :

1. Analysez le [corps](#) de [la réponse](#) de *navigationParams* en utilisant les règles pour les types en plusieurs parties. [\[RFC2046\]](#)
2. Soit *firstPartNavigationParams* une copie de *navigationParams*.
3. Définissez [la réponse](#) de *firstPartNavigationParams* sur une nouvelle [réponse](#) représentant la première partie du flux en plusieurs parties du [corps](#) de la [réponse de](#) *navigationParams*.
4. Soit *document* le résultat du [chargement d'un document](#) donné *firstPartNavigationParams*, *sourceSnapshotParams* et *initiatorOrigin*.

Pour chaque partie de corps supplémentaire obtenue à partir de [la réponse](#) de *navigationParams*, l'agent utilisateur doit [naviguer dans](#) le nœud du *document* [navigable](#) vers l' [URL](#) de la [requête](#) de *navigationParams*, en utilisant *document*, avec [la réponse](#) définie sur [la réponse](#) de *navigationParams* et [historyHandling](#) défini sur " ". [replace](#)

5. *Document* de retour.

Pour les besoins des algorithmes traitant ces parties de corps comme s'il s'agissait de ressources autonomes complètes, l'agent utilisateur doit agir comme s'il n'y avait plus d'octets pour ces ressources chaque fois que la limite suivant la partie de corps est atteinte.

Ainsi, [load](#) les événements (et d'ailleurs [unload](#) les événements) se déclenchent pour chaque partie du corps chargée.

### 7.5.6 Chargement de documents multimédias

Pour **charger un document multimédia**, donné *navigationParams* et un *type* de chaîne :



1. Soit *document* le résultat de [la création et de l'initialisation d'un Document objet](#) donné " *html*", *type* et *navigationParams* .
2. Réglez [le mode](#) du *document* sur " ".no-quirks
3. Ajouter un [html](#)élément au *document* .
4. Ajouter un [head](#)élément à l' [html](#)élément.
5. Ajouter un [body](#)élément à l' [html](#)élément.
6. Ajoutez un *élément host element* pour le média, comme décrit ci-dessous, à l' [body](#)élément.
7. Définissez l'attribut approprié de l'élément *host element* , comme décrit ci-dessous, sur l'adresse de la ressource image, vidéo ou audio.
8. Les agents utilisateurs peuvent ajouter du contenu à l' [head](#)élément de *document* , ou des attributs à l' *élément hôte* , par exemple, pour créer un lien vers une feuille de style, pour fournir un script, pour donner au document un [title](#)ou pour rendre le média [autoplay](#) .
9. [Traite les en-têtes de lien](#) donnés au *document* , [la réponse](#) de *navigationParams* et " ".media
10. Agir comme si l'agent utilisateur avait [arrêté d'analyser](#) *document* .
11. *Document* de retour .

L'élément *host element* à créer pour le média est l'élément donné dans le tableau ci-dessous dans la deuxième cellule de la ligne dont la première cellule décrit le média. L'attribut approprié à définir est celui donné par la troisième cellule de cette même ligne.

Type de média	Élément pour les médias	Attribut approprié
Image	<a href="#">img</a>	<a href="#">src</a>
Vidéo	<a href="#">video</a>	<a href="#">src</a>
l'audio	<a href="#">audio</a>	<a href="#">src</a>

Avant toute exécution de script, l'agent utilisateur doit attendre que [les scripts puissent s'exécuter pour que le document nouvellement créé](#) soit vrai pour le [Document](#).

### 7.5.7 Chargement d'un document pour le contenu en ligne qui n'a pas de DOM



Lorsque l'agent utilisateur doit créer un document pour afficher une page d'agent utilisateur ou un visualiseur PDF en ligne, fourni un *navigable* [navigable](#) , un [ID de navigation](#) *navigationId* , un *navTimingType* , l'agent utilisateur doit : [NavigationTimingType](#)

1. Soit *origin* une nouvelle [origine opaque](#) .
2. Laissez *coop* être une nouvelle [politique d'ouverture d'origine croisée](#) .
3. Soit *coopEnforcementResult* un nouveau [résultat d'application de la politique d'ouverture d'origine croisée](#) avec

[URL](#)

[URL](#) de la réponse

[origine](#)

*origine*

[politique d'ouverture d'origine croisée](#)

*coopérative*

4. Laissez *navigationParams* être un nouveau [paramètre de navigation](#) avec

[identifiant](#)

*ID de navigation*

[demande](#)

nul

[réponse](#)

une nouvelle [réponse](#)

[origine](#)

*origine*

[conteneur de stratégie](#)

un nouveau [conteneur de stratégie](#)

[ensemble final de drapeaux de sandboxing](#)

un ensemble vide

[politique d'ouverture d'origine croisée](#)

*coopérative*

[Résultat de l'application COOP](#)

*coopEnforcementResult*

[environnement réservé](#)

nul

[navigable](#)

*navigable*

[type de temps de navigation](#)

*navTimingType*

[récupérer le contrôleur](#)

récupérer le contrôleur

[valider les premiers indices](#)

nul

5. Soit *document* le résultat de [la création et de l'initialisation d'un Document objet](#) donné " *html*", " *text/html*" et *navigationParams* .

6. Associez *le document* à un rendu personnalisé qui n'est pas rendu à l'aide des Document règles de rendu normales ou faites muter *le document* jusqu'à ce qu'il représente le contenu que l'agent utilisateur souhaite rendre.
7. *Document* de retour .

*Étant donné que nous nous assurons que l' origineDocument du résultat est opaque et que le résultat ne peut pas exécuter de script avec accès au DOM, l'existence et les propriétés de celui-ci ne sont pas observables pour le code du développeur Web. Cela signifie que la plupart des valeurs ci-dessus, par exemple, le type , n'ont pas d'importance. De même, la plupart des éléments de navigationParams n'ont aucun effet observable, en plus d'empêcher l' algorithme -creation de se confondre, et sont donc définis sur des valeurs par défaut.*DocumentDocumenttext/html Document

Une fois la page configurée, l'agent utilisateur doit agir comme s'il avait arrêté d'analyser .

### 7.5.8 Terminer le processus de chargement

A Document a un **temps complètement chargé** (un temps ou nul), qui est initialement nul.

A Document est considéré comme **complètement chargé** si son temps complètement chargé est non nul.

Pour **terminer complètement le chargement** d'un Document document :

1. Assert : le contexte de navigation du *document* est non nul.
2. Définissez l'heure de chargement complet du *document* sur l'heure actuelle.
3. Soit *container* le conteneur du nœud navigable du *document* .

*Ce sera nul dans le cas où document est l' initialeabout:blank Document de a frame ou iframe, car au moment de la création du contexte de navigation qui appelle cet algorithme, la relation de conteneur n'est pas encore établie. (Cela se produit dans une étape ultérieure de création d'un nouvel enfant navigable .)*

*La conséquence en est que les étapes suivantes ne font rien, c'est-à-dire que nous ne déclenchons pas d' load événement asynchrone sur l'élément conteneur dans de tels cas. Au lieu de cela, un événement synchrone load est déclenché dans un cas spécial d'insertion initiale lors du traitement des iframe attributs .*

4. Si *conteneur* est un iframe élément, mettez en file d'attente une tâche d'élément sur la source de la tâche de manipulation

DOM *conteneur* donné pour exécuter les étapes de l'événement de chargement iframe *conteneur* donné .

5. Sinon, si *conteneur* n'est pas nul, mettez en file d'attente une tâche d'élément sur la source de tâche de manipulation DOM donnée *conteneur* pour déclencher un événement nommé load*conteneur* .

## 7.5.9 Déchargement des documents

A Document a un état **recupérable** , qui doit initialement être vrai, et une **page affichant** un indicateur, qui doit initialement être faux. L'indicateur de page montrant est utilisé pour s'assurer que les scripts reçoivent les événements pageshow et pagehide de manière cohérente (par exemple, qu'ils ne reçoivent jamais deux pagehide événements d'affilée sans l'intervention d'un pageshow, ou vice versa).

A Document a un DOMHighResTimeStamp **temps de suspension** , initialement 0.

A Document a une liste de **poignées de minuterie suspendues** , initialement vides.

Les boucles d'événements ont un compteur **de niveau d'imbrication de terminaison** , qui doit initialement être égal à 0.

Document les objets ont un **compteur de déchargement** , qui est utilisé pour ignorer certaines opérations pendant l'exécution des algorithmes ci-dessous. Initialement, le compteur doit être mis à zéro.

Pour **décharger** un Document *oldDocument* , étant donné un Document *newDocument* facultatif :

1. Assert : cela s'exécute dans le cadre d'une tâche en file d'attente sur la boucle d'événements de *oldDocument* .
2. Soit *unloadTimingInfo* une nouvelle information de temps de déchargement de document .
3. Si *newDocument* n'est pas donné, définissez *unloadTimingInfo* sur null.

*Dans ce cas, aucun nouveau document n'a besoin de savoir combien de temps il a fallu à oldDocument pour se décharger.*

4. Sinon, si la boucle d'événements de *newDocument* n'est pas la boucle d'événements de *oldDocument* , alors l'agent utilisateur peut décharger *oldDocument* en parallèle . Dans ce cas, l'agent utilisateur doit définir *unloadTimingInfo* sur null.

*Dans ce cas, le chargement de newDocument n'est pas affecté par le temps qu'il faut pour décharger oldDocument , il serait donc inutile de communiquer ces informations de synchronisation.*

5. Soit *intentToStoreInBfcache* vrai si l'agent utilisateur a l'intention de garder *oldDocument* actif dans une [entrée d'historique de session](#) , de sorte qu'il puisse être [utilisé ultérieurement pour parcourir l'historique](#) .

Cela doit être faux si *oldDocument* n'est pas [récupérable](#) , ou s'il existe des descendants de *oldDocument* que l'agent utilisateur n'a pas l'intention de maintenir en vie de la même manière (y compris en raison de leur manque de [récupérable](#) ).

6. Soit *eventLoop* la [boucle d'événements](#) de [l'agent correspondant](#) de *oldDocument* .
7. Augmentez [le niveau d'imbrication de terminaison](#) de *eventLoop* de 1.
8. Augmentez [le compteur de déchargement](#) de *oldDocument* de 1.
9. Si *intentToKeepInBfcache* est faux, alors définissez l'état [récupérable](#) de *oldDocument* sur faux.
10. Si [l'affichage de la page](#) de *oldDocument* est vrai :
  1. Définissez [l'affichage](#) de la page de *oldDocument* sur false.
  2. [Déclenchez un événement de transition de page](#) nommé `pagehide` sur [l'objet global pertinent](#) de *oldDocument* avec l'état [récupérable](#) de *oldDocument* .
  3. [Mettez à jour l'état de visibilité](#) de *oldDocument* sur " `hidden` ".
11. Si *unloadTimingInfo* n'est pas nul, définissez [l'heure de début de l'événement de déchargement](#) de *unloadTimingInfo* sur l' [heure haute résolution actuelle](#) en fonction de l' [objet global pertinent](#) de *newDocument* , `grossi` compte tenu de [la capacité d'isolement cross-origin](#) de [l'objet de paramètres pertinents](#) de *oldDocument* .
12. Si l'état [récupérable](#) de *oldDocument* est faux, [déclenchez un événement](#) nommé sur [l'objet global pertinent](#) de *oldDocument* , avec l'indicateur de remplacement de cible hérité défini `unload`.
13. Si *unloadTimingInfo* n'est pas nul, définissez [l'heure de fin de l'événement de déchargement](#) de *unloadTimingInfo* sur l' [heure haute résolution actuelle](#) en fonction de l' [objet global pertinent](#) de *newDocument* , `grossi` compte tenu de [la capacité d'isolement d'origine croisée](#) de [l'objet de paramètres pertinent](#) de *oldDocument* .
14. Diminuez [le niveau d'imbrication de terminaison](#) de *eventLoop* de 1.

15. Définissez le temps de suspension de *oldDocument* sur le temps haute résolution actuel donné à l'objet global pertinent du *document* .
16. Définissez les descripteurs de temporisateurs suspendus de *oldDocument* sur le résultat de l'obtention des clés pour la carte des temporisateurs actifs .
17. Set *oldDocument* 's has been scrolled by the user to false.
18. Exécutez toutes les étapes de nettoyage de document de déchargement pour *oldDocument* qui sont définies par cette spécification et d'autres spécifications applicables .
19. Si l'état récupérable de *oldDocument* est faux, alors détruisez *oldDocument* .
20. Diminue le compteur de déchargement de *oldDocument* de 1.
21. Si *newDocument* est donné, *newDocument* a été créé via des redirections cross-origin est false, et l'origine de *newDocument* est la même que l'origine de *oldDocument* , puis définissez le délai de déchargement du document précédent de *newDocument* sur *unloadTimingInfo* .

Cette spécification définit les **étapes suivantes de nettoyage des documents lors du déchargement** . D'autres spécifications peuvent en définir davantage. Donné un Document *document* :

1. Soit *window* l' objet global pertinent du *document* .
2. Pour chaque WebSocket objet *webSocket* dont l'objet global pertinent est *window* , faites disparaître *webSocket* .  
  
Si cela a affecté des WebSocket objets, définissez l'état récupérable du *document* sur false.
3. Si l'état récupérable du *document* est faux, alors :
  1. Pour chaque EventSource objet *eventSource* dont l'objet global pertinent est égal à *window* , fermez de force *eventSource* .
  2. Effacer la carte de *la fenêtre* des minuteries actives .

### 7.5.10 Destruction de documents

Pour **détruire** un Document *document* :

1. Détruit les documents actifs de chacun des navigables descendants du *document* . Dans quel ordre?

2. Définissez l'état recupérable du *document* sur false.
3. Exécutez toutes les étapes de nettoyage de document de téléchargement pour le *document* défini par cette spécification et d'autres spécifications applicables .
4. Abandonner le *document* .
5. Supprimez toutes les tâches dont le *document* est *document* de toute file d'attente de tâches (sans exécuter ces tâches).
6. Définissez le contexte de navigation du *document* sur null.
7. Définissez l'état du *document* de l'entrée d'historique de session active du nœud navigable du *document* sur null.
8. Supprime *document* du jeu propriétaire de chaque WorkerGlobalScope objet dont le jeu contient *document* .
9. Pour chaque *workletGlobalScope* dans les étendues globales de worklet du *document* , terminez *workletGlobalScope* .

*Même après la destruction, l' Document objet lui-même peut toujours être accessible au script, dans le cas où nous détruisons un enfant navigable .*

### 7.5.11 Abandon du chargement d'un document

Pour **abandonner** un Document *document* :

1. Abandonner les documents actifs de chacun des navigables descendants du *document* . **Dans quel ordre?** S'il en résulte que l'état recupérable de l'un de ces objets est défini sur faux, définissez également l'état recupérable du *document* sur faux. Document
2. Annulez toutes les instances de l' algorithme de récupération dans le contexte de *document* , en supprimant toutes les tâches mises en file d'attente pour elles et en supprimant toutes les données supplémentaires reçues du réseau pour elles. Si cela a entraîné l'annulation d'instances de l' algorithme de récupération ou l'annulation de tâches en file d'attente ou de données réseau, définissez l' état recupérable du *document* sur false.
3. Si l'identifiant de navigation du *document* n'est pas nul, alors :
  1. Appelez la navigation WebDriver BiDi abandonnée avec le contexte de navigation du *document* et le nouveau statut de navigation WebDriver BiDi dont l'ID est l'ID de navigation du *document* , le statut est " " et l'URL est l'URL du *document* . canceled

2. Définissez [l'ID de navigation](#) du *document* sur null.
4. Si le *document* a un [analyseur actif](#) , alors :
  1. L'analyseur actif du *document* défini [a été abandonné](#) sur true.
  2. [Abandonner cet analyseur](#) .
  3. Définissez l'état [récupérable](#) du *document* sur false.

Pour **arrêter le chargement** d'un *navigable* [navigable](#) :

1. Soit *document* le [document actif](#) de *navigable* .
2. Si [le compteur de déchargement](#) du *document* est 0 et que [la navigation en cours](#) de *navigable* est un [ID de navigation](#) , alors définissez [la navigation en cours de](#) *navigable* sur null.

*Cela aura pour effet d'interrompre toute navigation en cours de navigable , car à certains moments de la navigation, des modifications de la [navigation en cours](#) entraîneront l'abandon de travaux supplémentaires.*

3. [Abandonner](#) le *document* .

Grâce à leur [interface utilisateur](#) , les agents utilisateurs permettent également d'arrêter les traversées, c'est-à-dire les cas où la [navigation en cours](#) est " traversal". L'algorithme ci-dessus ne tient pas compte de cela. (D'un autre côté, les agents utilisateurs ne permettent pas `window.stop()` d'arrêter les traversées, donc l'algorithme ci-dessus est correct pour cet appelant.) Voir [le problème #6905](#) .

## 7.6 L' [X-Frame-Options](#) en-tête ``



L' [X-Frame-Options](#) en - tête de réponse HTTP `` est un moyen hérité de contrôler si et comment un [Document](#) peut être chargé à l' intérieur d' un [enfant navigable](#) . Elle est rendue obsolète par la [frame-ancestors](#) directive CSP, qui offre un contrôle plus granulaire sur les mêmes situations. Il a été défini à l'origine dans *HTTP Header Field X-Frame-Options* , mais la définition et le modèle de traitement ici remplacent ce document. [\[CSP\]](#) [\[RFC7034\]](#)

*En particulier, HTTP Header Field X-Frame-Options a spécifié une `ALLOW-FROM` variante `` de l'en-tête, mais cela ne doit pas être implémenté. Selon le modèle de traitement ci-dessous, si une [frame-ancestors](#) directive CSP et un [X-Frame-Options](#) en-tête `` sont utilisés dans la même [réponse](#) , alors `` [X-Frame-Options](#) `` est ignoré.*



Pour les développeurs Web et les vérificateurs de conformité, sa valeur [ABNF](#) est :

```
X-Frame-Options = "DENY" / "SAMEORIGIN"
```

Pour **vérifier l'adhésion d'une réponse de navigation** à `X-Frame-Options`, étant donné une [réponse](#) `response`, un [navigable](#) `navigable`, une [liste CSP](#) `cspList` et une [origine](#) `destinationOrigin` :

1. Si `navigable` n'est pas un [enfant navigable](#), alors retourne `true`.
2. [Pour chaque](#) `politique` de `cspList` :
  1. Si [la disposition](#) de la `politique` n'est pas " ", alors [continuez](#) `.enforce`
  2. Si [le jeu de directives](#) de la `stratégie` [contient](#) une directive, alors renvoie `true`. `frame-ancestors`
3. Soit `rawXFrameOptions` le résultat de [l'obtention, du décodage et de la séparation de](#) `X-Frame-Options` de [la liste](#) d'en-tête de la `réponse`.
4. Soit `xFrameOptions` un nouvel [ensemble](#).
5. [Pour chaque](#) valeur de `rawXFrameOptions`, [ajoutez](#) `value`, [convertie en minuscules ASCII](#), à `xFrameOptions`.
6. Si [la taille](#) de `xFrameOptions` est supérieure à 1 et que `xFrameOptions` [contient](#) l'un des " ", " " ou " ", renvoie `false`. `denyallowallsameorigin`

*L'intention ici est de bloquer toute tentative d'application de `X-Frame-Options` qui tentait de faire quelque chose de valable, mais qui semble confuse.  
Il s'agit du seul impact de la valeur héritée `ALLOWALL` sur le modèle de traitement.*

7. Si [la taille](#) de `xFrameOptions` est supérieure à 1, renvoie `true`.

*Cela signifie qu'il contient plusieurs valeurs non valides, que nous traitons de la même manière que si l'en-tête était entièrement omis.*

8. Si `xFrameOptions` [0] vaut " `deny`", alors renvoie `false`.
9. Si `xFrameOptions` [0] est " `sameorigin`", alors :
  1. Soit `containerDocument` le [document conteneur](#) de `navigable`.
  2. [Alors que](#) `containerDocument` n'est pas `null` :
    1. Si [l'origine](#) de `containerDocument` n'est pas [la même origine](#) que `destinationOrigin`, alors renvoyez `false`.



2. Définissez `containerDocument` sur [le document conteneur](#) de `containerDocument`.

10. Renvoie vrai.

*Si nous avons atteint ce point, nous avons une seule valeur invalide (qui pourrait potentiellement être l'une des formes héritées `ALLOWALL`` ou `ALLOW-FROM``). Ceux-ci sont traités comme si l'en-tête était entièrement omis.*

Le tableau suivant illustre le traitement des différentes valeurs de l'en-tête, y compris celles non conformes :

<code>`X-Frame-Options`</code>	Valide	Résultat
<code>`DENY`</code>	✓	intégration non autorisée
<code>`SAMEORIGIN`</code>	✓	intégration de même origine autorisée
<code>`INVALID`</code>	✗	intégration autorisée
<code>`ALLOWALL`</code>	✗	intégration autorisée
<code>`ALLOW-FROM=https://example.com/`</code>	✗	intégration autorisée (de n'importe où)

Le tableau suivant illustre le traitement de divers cas non conformes impliquant plusieurs valeurs :

<code>`X-Frame-Options`</code>	Résultat
<code>`SAMEORIGIN, SAMEORIGIN`</code>	intégration de même origine autorisée
<code>`SAMEORIGIN, DENY`</code>	intégration non autorisée
<code>`SAMEORIGIN, `</code>	intégration non autorisée
<code>`SAMEORIGIN, ALLOWALL`</code>	intégration non autorisée
<code>`SAMEORIGIN, INVALID`</code>	intégration non autorisée
<code>`ALLOWALL, INVALID`</code>	intégration non autorisée
<code>`ALLOWALL, `</code>	intégration non autorisée
<code>`INVALID, INVALID`</code>	intégration autorisée

Les mêmes résultats sont obtenus que les valeurs soient délivrées dans un seul en-tête dont la valeur est délimitée par des virgules, ou dans plusieurs en-têtes.

## 7.7 L' Refresh en-tête ``

L' **Refresh**-tête de réponse HTTP `` est l'équivalent HTTP d'un **meta**élément avec un **http-equiv** attribut dans l' **état Refresh** . Il prend **la même valeur** et fonctionne en grande partie de la même manière. Son modèle de traitement est détaillé dans **créer et initialiser un Document objet** .

## 7.8 Considérations relatives à l'interface utilisateur du navigateur

Les agents utilisateurs de navigateur doivent fournir la possibilité de **naviguer** , **de recharger** et **d'arrêter de charger** tout **traversable de niveau supérieur** dans leur **ensemble traversable de niveau supérieur** .

Par exemple, via une barre d'emplacement et une interface utilisateur de bouton recharger/arrêter.

Les agents utilisateurs de navigateur devraient fournir la possibilité de **traverser par un delta** tout **traversable de niveau supérieur** dans leur **ensemble traversable de niveau supérieur** .

Par exemple, via les boutons Précédent et Suivant, y compris éventuellement des capacités d'appui long pour modifier le delta.

Il est suggéré que de tels agents utilisateurs autorisent le parcours par des deltas supérieurs à un, pour éviter de laisser une page "piéger" l'utilisateur en bourrant l'historique de session avec des entrées erronées. (Par exemple, via des appels répétés à **history.pushState()** ou **des navigations fragmentées** .)

*Certains agents utilisateurs ont des heuristiques pour traduire une simple pression sur un bouton "précédent" ou "suivant" en un delta plus grand, spécifiquement pour surmonter de tels abus. Nous envisageons de préciser ces heuristiques dans **le numéro 7832** .*

Les agents utilisateurs du navigateur doivent offrir aux utilisateurs la possibilité de **créer un nouveau parcours de niveau supérieur** , à partir d'une **URL** initiale fournie par l'utilisateur ou déterminée par l'agent utilisateur .

Par exemple, via un bouton "nouvel onglet" ou "nouvelle fenêtre".

Les agents utilisateurs de navigateur devraient offrir aux utilisateurs la possibilité de **fermer** arbitrairement tout **traversable de niveau supérieur** dans leur **ensemble traversable de niveau supérieur** .

Par exemple, en cliquant sur un bouton "fermer l'onglet".

---

Les agents utilisateurs du navigateur peuvent fournir des moyens à l'utilisateur de provoquer explicitement [la navigation](#) , [le rechargement](#) ou [l'arrêt du chargement de n'importe quel navigable](#) (pas seulement un [traversable de niveau supérieur](#) ) .

Par exemple, via un menu contextuel.

Les agents utilisateurs du navigateur peuvent permettre aux utilisateurs de [détruire un fichier traversable de niveau supérieur](#) .

Par exemple, en fermant de force une fenêtre contenant un ou plusieurs de ces [traversables de niveau supérieur](#) .

---

Lorsqu'un utilisateur demande le [rechargement](#) d'un [élément navigable](#) dont [la ressource](#) de l' état du [document de l'entrée d'historique de session active](#) est une [ressource POST](#) , l'agent utilisateur doit d'abord inviter l'utilisateur à confirmer l'opération, sinon les transactions (par exemple, les achats ou les modifications de la base de données) pourrait être répété.

Lorsqu'un utilisateur demande le [rechargement](#) d'un [navigable](#) , les agents utilisateurs peuvent fournir un mécanisme pour ignorer les caches lors du rechargement.

---

Les recommandations ci-dessus, et les structures de données de cette spécification, ne sont pas destinées à imposer des restrictions sur la façon dont les agents utilisateurs représentent l'historique de session à l'utilisateur.

Par exemple, bien que [les entrées d'historique de session](#) d' [un traversable de niveau supérieur](#) soient stockées et maintenues sous forme de liste, et qu'il soit recommandé à l'agent utilisateur de donner une interface pour [parcourir cette liste par un delta](#) , un nouvel agent utilisateur pourrait à la place ou en plus présenter une vue arborescente, chaque page ayant plusieurs pages "en avant" entre lesquelles l'utilisateur peut choisir.

De même, bien que l'historique de session pour tous [les navigables](#) descendants soit stocké dans leur [traversable navigable](#) , les agents utilisateurs pourraient présenter à l'utilisateur une vue [pernavigable](#) plus nuancée de l'historique de session.

---

Les agents utilisateurs du navigateur peuvent utiliser un contexte [de navigation de niveau supérieur is popup](#) boolean aux fins suivantes :

- Décider de fournir ou non une interface utilisateur de navigateur Web minimale pour le [traversable de niveau supérieur](#) correspondant .
- Exécution des étapes facultatives de [configuration des fonctionnalités de contexte de navigation](#) .

Dans les deux cas, les agents utilisateurs peuvent en outre incorporer les préférences de l'utilisateur ou présenter le choix de suivre ou non la route contextuelle.

Les agents utilisateurs qui fournissent une interface utilisateur minimale pour ces fenêtres contextuelles sont encouragés à ne pas masquer la barre d'emplacement du navigateur.

## 1. [8 API d'applications Web](#)

### 1. [8.1 Script](#)

#### 1. [8.1.1 Présentation](#)

#### 2. [8.1.2 Agents et grappes d'agents](#)

##### 1. [8.1.2.1 Intégration avec le formalisme de l'agent JavaScript](#)

##### 2. [8.1.2.2 Intégration avec le formalisme du cluster d'agents JavaScript](#)

#### 3. [8.1.3 Les royaumes et leurs homologues](#)

##### 1. [8.1.3.1 Environnements](#)

##### 2. [8.1.3.2 Objets de paramètres d'environnement](#)

##### 3. [8.1.3.3 Domaines, objets de paramètres et objets globaux](#)

###### 1. [8.1.3.3.1 Saisie](#)

###### 2. [8.1.3.3.2 Titulaire](#)

###### 3. [8.1.3.3.3 Courant](#)

###### 4. [8.1.3.3.4 Pertinent](#)

##### 4. [8.1.3.4 Activation et désactivation des scripts](#)

##### 5. [8.1.3.5 Contextes sécurisés](#)

#### 4. [8.1.4 Modèle de traitement de script](#)

##### 1. [8.1.4.1 Scénarios](#)

##### 2. [8.1.4.2 Récupérer des scripts](#)

##### 3. [8.1.4.3 Création de scripts](#)

##### 4. [8.1.4.4 Appel de scripts](#)

##### 5. [8.1.4.5 Arrêter des scripts](#)

##### 6. [8.1.4.6 Erreurs de script d'exécution](#)

##### 7. [8.1.4.7 Refus de promesses non gérées](#)

##### 8. [8.1.4.8 Importer les résultats d'analyse de carte](#)

#### 5. [8.1.5 Résolution du spécificateur de module](#)

##### 1. [8.1.5.1 L'algorithme de résolution](#)

##### 2. [8.1.5.2 Importer des cartes](#)

##### 3. [8.1.5.3 Importer le modèle de traitement de carte](#)

#### 6. [8.1.6 Crochets d'hôte de spécification JavaScript](#)

##### 1. [8.1.6.1 HostEnsureCanAddPrivateElement\( \*O\* \)](#)

##### 2. [8.1.6.2 HostEnsureCanCompileStrings\( \*domaine\* \)](#)

##### 3. [8.1.6.3 HostPromiseRejectionTracker\( \*promesse\*, \*opération\* \)](#)

##### 4. [8.1.6.4 Crochets d'hôte liés à la tâche](#)

###### 1. [8.1.6.4.1 HostCallJobCallback\( \*callback\*, \*V\*, \*argumentsList\* \)](#)

###### 2. [8.1.6.4.2 HostEnqueueFinalizationRegistryCleanupJob\( \*finalizationRegistry\* \)](#)

3. [8.1.6.4.3 HostEnqueuePromiseJob\( \*travail\*, \*domaine\* \)](#)
4. [8.1.6.4.4 HostMakeJobCallback\( \*appellable\* \)](#)
5. [8.1.6.5 Crochets d'hôte liés au module](#)
  1. [8.1.6.5.1 HostGetImportMetaProperties\( \*moduleRecord\* \)](#)
  2. [8.1.6.5.2 HostGetSupportedImportAssertions\(\)](#)
  3. [8.1.6.5.3 HostLoadImportedModule\( \*referrer\*, \*moduleRequest\*, \*loadState\*, \*payload\* \)](#)
7. [8.1.7 Boucles d'événements](#)
  1. [8.1.7.1 Définitions](#)
  2. [8.1.7.2 Mise en file d'attente des tâches](#)
  3. [8.1.7.3 Modèle de traitement](#)
  4. [8.1.7.4 Sources de tâches génériques](#)
  5. [8.1.7.5 Traitement de la boucle d'événements à partir d'autres spécifications](#)
8. [8.1.8 Événements](#)
  1. [8.1.8.1 Gestionnaires d'événements](#)
  2. [8.1.8.2 Gestionnaires d'événements sur des éléments, \*Document\* des objets et \*Window\* des objets](#)
    1. [8.1.8.2.1 Définitions IDL](#)
  3. [8.1.8.3 Déclenchement d'événement](#)
2. [8.2 Le \*WindowOrWorkerGlobalScope\* mixage](#)
3. [8.3 Méthodes utilitaires Base64](#)

## 8 API d'applications Web

### 8.1 Script

#### 8.1.1 Présentation

Divers mécanismes peuvent entraîner l'exécution d'un code exécutable fourni par l'auteur dans le contexte d'un document. Ces mécanismes incluent, mais ne sont probablement pas limités à :

- Traitement des [script](#) éléments.
- Navigation vers [javascript:les URL](#) .
- Les gestionnaires d'événements, qu'ils soient enregistrés via le DOM à l'aide de `addEventListener()` , par [des attributs de contenu de gestionnaire](#)

[d'événements](#) explicites , par [des attributs IDL de gestionnaire d'événements](#) ou autrement.

- Traitement de technologies comme SVG qui ont leurs propres fonctionnalités de script.

## 8.1.2 Agents et grappes d'agents

### 8.1.2.1 Intégration avec le formalisme de l'agent JavaScript

JavaScript définit le concept d'un [agent](#) . Cette section donne le mappage de ce concept au niveau du langage sur la plate-forme Web.

*Conceptuellement, le concept [d'agent](#) est un "thread" idéalisé indépendant de l'architecture dans lequel le code JavaScript s'exécute. Un tel code peut impliquer plusieurs globals/ [domaines](#) qui peuvent accéder les uns aux autres de manière synchrone et doivent donc s'exécuter dans un seul thread d'exécution.*

*Deux [Window](#) objets ayant le même [agent](#) n'indiquent pas qu'ils peuvent accéder directement à tous les objets créés dans les domaines de l'autre. Ils devraient être [du même domaine d'origine](#) ; voir [IsPlatformObjectSameOrigin](#) .*

Les types d'agents suivants existent sur la plateforme Web :

#### **Agent de fenêtre d'origine similaire**

Contient divers [Window](#) objets qui peuvent potentiellement s'atteindre, soit directement, soit en utilisant [document.domain](#).

Si la [clé d'origine](#) du [cluster d'agents](#) englobant est vraie, alors tous les objets auront [la même origine](#) , pourront s'atteindre directement et ne fonctionneront pas. [Window.document.domain](#)

*Deux [Window](#) objets de [même origine](#) peuvent se trouver dans différents [agents de fenêtre d'origine similaire](#) , par exemple s'ils se trouvent chacun dans leur propre [groupe de contexte de navigation](#) .*

#### **Agent travailleur dédié**

Contient un seul [DedicatedWorkerGlobalScope](#).

#### **Agent de travail partagé**

Contient un seul [SharedWorkerGlobalScope](#).

#### **Agent de service**

Contient un seul [ServiceWorkerGlobalScope](#).

#### **Agent Worklet**

Contient un seul [WorkletGlobalScope](#) objet.

*Bien qu'un worklet donné puisse avoir plusieurs domaines, chacun de ces domaines a besoin de son propre agent, car chaque domaine peut exécuter du code indépendamment et en même temps que les autres.*

Seuls les agents de travail [partagés](#) et [dédiés](#) autorisent l'utilisation d' [Atoms](#) API JavaScript pour potentiellement [bloquer les fichiers](#) .

Pour **créer un agent** , étant donné un booléen *canBlock* :

1. Soit *signifiant* une nouvelle valeur interne unique.
2. Soit *candidateExecution* une nouvelle [exécution candidate](#) .
3. Soit *agent* un nouvel [agent](#) dont `[[CanBlock]]` est *canBlock* , `[[Signifier]]` est *signifiant* , `[[CandidateExecution]]` est *candidateExecution* et `[[IsLockFree1]]`, `[[IsLockFree2]]` et `[[LittleEndian]]` sont fixés à la discrétion de l'implémentation.
4. Définissez [la boucle d'événements](#) de l' *agent* sur une nouvelle [boucle d'événements](#) .
5. *Agent* de retour .

L' **agent pertinent** pour un [objet de plate-forme](#) *platformObject* est l'[agent](#) du [domaine pertinent](#) de *platformObject* . **Ce**  
pointeur n'est pas encore défini dans la spécification JavaScript  
; voir [tc39/ecma262#1357](#) .

*L'agent équivalent du [domaine actuel](#) est l' [agent environnant](#) .*

### 8.1.2.2 Intégration avec le formalisme du cluster d'agents JavaScript

JavaScript définit également le concept de [cluster d'agents](#) , que cette norme mappe à la plate-forme Web en plaçant les agents de manière appropriée lorsqu'ils sont créés à l'aide des algorithmes [d'obtention d'un agent de fenêtre d'origine similaire](#) ou [d'obtention d'un agent de travail/worklet](#) .

Le concept [de cluster d'agents](#) est crucial pour définir le modèle de mémoire JavaScript, et en particulier entre quels [agents](#) les données de sauvegarde des [SharedArrayBuffer](#) objets peuvent être partagées.

*Conceptuellement, le concept [de cluster d'agents](#) est une « frontière de processus » idéale, indépendante de l'architecture, qui regroupe plusieurs « threads » ( [agents](#) ). Les [clusters d'agents](#) définis par la spécification sont généralement plus restrictifs que les frontières de processus réelles mises en œuvre dans les agents*



*utilisateurs. En appliquant ces divisions idéalisées au niveau des spécifications, nous nous assurons que les développeurs Web voient un comportement interopérable en ce qui concerne la mémoire partagée, même face à des modèles de processus d'agent utilisateur variables et changeants.*

Un [cluster d'agents](#) est associé à un **mode d'isolation cross-origin** , qui est un [mode d'isolation cross-origin](#) . C'est initialement " [none](#) ".

Un [cluster d'agents](#) est associé à **une clé d'origine** (un booléen), qui est initialement faux.

---

Ce qui suit définit l'allocation des [clusters d'agents](#) d'agents de [fenêtre d'origine similaire](#) .

Une **clé de cluster d'agents** est une [origine de site](#) ou de tuple . Sans l'action du développeur Web pour réaliser [des clusters d'agents à clé d'origine](#) , ce sera un [site](#) .

*Une formulation équivalente est qu'une [clé de cluster d'agent](#) peut être un [schema-and-host](#) ou un [origin](#) .*

Pour **obtenir un agent de fenêtre d'origine similaire** , en fonction d'une *origine* [origin](#) , d'un *groupe* [de groupes de contextes de navigation](#) et d'un booléen *requestsOAC* , exécutez ces étapes :

1. Soit *site* le résultat de [l'obtention d'un site](#) avec *origin* .
2. Laissez *la clé* être *le site* .
3. Si [le mode d'isolation cross-origin](#) du *groupe* n'est pas " " , alors définissez *key* sur *origin* [.none](#) .
4. Sinon, si [la mappe de clés du cluster d'agents historique](#) du *groupe* [ *origine* ] [existe](#) , définissez *la clé* sur [la mappe de clés du cluster d'agents historique](#) du *groupe* [ *origine* ] .
5. Sinon:
  1. Si *requestsOAC* est vrai, définissez *la clé* sur *origin* .
  2. Définissez [le mappage de clés du cluster d'agents historique](#) du *groupe* [ *origine* ] sur *key* .
6. Si [la carte du cluster d'agents](#) du *groupe* [ *clé* ] [n'existe pas](#) , alors :
  1. Soit *agentCluster* un nouveau [cluster d'agents](#) .

2. Définissez [le mode d'isolation cross-origin](#) de *l'agentCluster* sur [le mode d'isolation cross-origin](#) du groupe .
3. Set *agentCluster* est [origin-keyed](#) à true si *key* est égal à *origin* ; sinon faux.
4. Ajoutez le résultat de [la création d'un agent](#) , donné false, à *agentCluster* .
5. Définissez [ *clé* ] [la carte du cluster d'agents](#) du groupe sur *agentCluster* .
7. Renvoie l'unique [agent de fenêtre d'origine similaire](#) contenu dans [la carte de cluster d'agents](#) du groupe [ *clé* ].

*Cela signifie qu'il n'y a qu'un seul [agent de fenêtre d'origine similaire](#) par grappe d'agents de contexte de navigation. (Cependant, [les agents](#) de travail et de worklet [dédiés](#) peuvent se trouver dans le même cluster.)*

---

Ce qui suit définit l'allocation des [clusters d'agents](#) de tous les autres types d'agents.

Pour **obtenir un agent de travail/worklet** , étant donné un [objet de paramètres d'environnement](#) ou des *paramètres extérieurs* nuls , un booléen *isTopLevel* et un booléen *canBlock* , exécutez ces étapes :

1. Laissez *agentCluster* être nul.
2. Si *isTopLevel* est vrai, alors :
  1. Définissez *agentCluster* sur un nouveau [cluster d'agents](#) .
  2. Définir [l'origine](#) de *l'agentCluster* est défini sur true.

*Ces travailleurs peuvent être considérés comme étant liés à l'origine. Cependant, cela n'est exposé par aucune API (de la manière qui [originAgentCluster](#) expose la clé d'origine pour Windows).*

3. Sinon:
  1. [Assert](#) : les *paramètres extérieurs* ne sont pas nuls.
  2. Laissez *ownerAgent* être en dehors de l'agent du [domaine](#) de *settings* .
  3. Définissez *agentCluster* sur le cluster d'agents qui contient *ownerAgent* .
4. Soit *agent* le résultat de [la création d'un agent](#) donné *canBlock* .

5. Ajouter un agent à `agentCluster` .

6. Agent de retour .

Pour **obtenir un agent de travail dédié/partagé** , étant donné un [objet de paramètres d'environnement](#) *en dehors des paramètres* et un booléen `isShared` , renvoyez le résultat de [l'obtention d'un agent de travail/worklet](#) donné *en dehors des paramètres* , `isShared` et `true`.

Pour **obtenir un agent de worklet** , étant donné un [objet de paramètres d'environnement](#) *en dehors de settings* , renvoyez le résultat de [l'obtention d'un agent de travail/worklet](#) donné *en dehors de settings* , `false` et `false`.

Pour **obtenir un agent de travail de service** , renvoyez le résultat de [l'obtention d'un agent de travail/worklet](#) avec `null`, `true` et `false`.

---

Les paires d'objets globaux suivantes se trouvent chacune dans le même [cluster d'agents](#) et peuvent donc utiliser [SharedArrayBuffer](#) des instances pour partager de la mémoire entre elles :

- Un [Window](#) objet et un travailleur dédié qu'il a créé.
- Un travailleur (de n'importe quel type) et un travailleur dédié qu'il a créé.
- Un [Window](#) objet A et l' [Window](#) objet d'un [iframe](#) élément créé *par A* qui pourrait être [le même domaine d'origine](#) que A .
- Un [Window](#) objet et un [même Window](#) objet de domaine d'origine qui l'a ouvert.
- Un [Window](#) objet et un worklet qu'il a créés.

Les paires d'objets globaux suivantes ne se trouvent *pas* dans le même [cluster d'agents](#) et ne peuvent donc pas partager de mémoire :

- Un [Window](#) objet et un travailleur partagé qu'il a créé.
- Un nœud de calcul (de n'importe quel type) et un nœud de calcul partagé qu'il a créé.
- Un [Window](#) objet et un service worker qu'il a créé.
- Un [Window](#) objet et l' [Window](#) objet d'un [iframe](#) élément créé *par A* qui ne peut pas être [le même domaine d'origine](#) que A .
- Deux [Window](#) objets quelconques sans relation d'ouverture ou d'ancêtre. Ceci est valable même si les deux [Window](#) objets ont [la même origine](#) .

### 8.1.3 Les royaumes et leurs homologues

La spécification JavaScript introduit le concept [de domaine](#) , représentant un environnement global dans lequel le script est exécuté. Chaque domaine est livré avec un [objet global défini par l'implémentation](#) ; une grande partie de cette spécification est consacrée à la définition de cet objet global et de ses propriétés.

Pour les spécifications Web, il est souvent utile d'associer des valeurs ou des algorithmes à une paire domaine/objet global. Lorsque les valeurs sont spécifiques à un type particulier de domaine, elles sont directement associées à l'objet global en question, par exemple dans la définition des interfaces [Window](#) ou [WorkerGlobalScope](#). Lorsque les valeurs ont une utilité dans plusieurs domaines, nous utilisons le concept [d'objet de paramètres d'environnement](#) .

Enfin, dans certains cas, il est nécessaire de suivre les valeurs associées avant même qu'un objet domaine/objet global/paramètres d'environnement n'existe (par exemple, pendant la [navigation](#) ). Ces valeurs sont suivies dans le concept [d'environnement](#) .

### 8.1.3.1 Environnements

Un **environnement** est un objet qui identifie les paramètres d'un environnement d'exécution actuel ou potentiel. Un [environnement](#) contient les champs suivants :

#### Un *identifiant*

Une chaîne opaque qui identifie de manière unique cet [environnement](#) .

#### Une *URL de création*

Une [URL](#) qui représente l'emplacement de la ressource à laquelle cet [environnement](#) est associé.

*Dans le cas d'un [Window](#) [objet de paramètres d'environnement](#) , cette URL peut être distincte de l'URL associée à son objet [global](#) , en [raison](#) de mécanismes tels que [qui modifient cette dernière](#). `Document.history.pushState()`*

#### Une *URL de création de premier niveau*

Null ou une [URL](#) qui représente l' [URL de création de l' environnement](#) "de niveau supérieur" . Elle est nulle pour les travailleurs et les worklets.

#### Une *origine de haut niveau*

**Pour l'instant** une valeur [définie par l'implémentation](#) , null, ou un [origin](#) . Pour un environnement d'exécution potentiel "de niveau supérieur", il est nul (c'est-à-dire lorsqu'il n'y a pas encore de réponse) ; sinon, il s'agit de l'[origine](#) de l' [environnement](#) "de niveau supérieur" . Pour un worker ou un worklet dédié, il s'agit de l' [origine de niveau supérieur](#) de son créateur. Pour un travailleur partagé ou de service, il s'agit d'une valeur [définie par l'implémentation](#) .

*Ceci est distinct de [l'origine de l' URL de création de niveau supérieur](#) lorsque le sandboxing, les nœuds de calcul et les worklets sont impliqués.*

#### **Un contexte de navigation cible**

Null ou un [contexte de navigation](#) cible pour une [requête de navigation](#) .

#### **Un travailleur de service actif**

Null ou un [service worker](#) qui [contrôle](#) l' [environnement](#) .

#### **Un indicateur prêt à l'exécution**

Un indicateur qui indique si la configuration de l'environnement est terminée. Il est initialement non défini.

Les spécifications peuvent définir **des étapes de suppression d'environnement** pour les environnements. Les étapes prennent un [environnement](#) comme entrée.

*Les [étapes de suppression d'environnement](#) ne sont exécutées que pour quelques environnements sélectionnés : ceux qui ne seront jamais prêts à être exécutés car, par exemple, ils n'ont pas pu être chargés.*

### **8.1.3.2 Objets de paramètres d'environnement**

Un **objet de paramètres d'environnement** est un [environnement](#) qui spécifie en outre des algorithmes pour :

#### **Un contexte d'exécution de domaine**

Un [contexte d'exécution JavaScript](#) partagé par tous [les scripts](#) qui utilisent cet objet de paramètres, c'est-à-dire tous les scripts d'un [domaine](#) donné . Lorsque nous [exécutons un script classique](#) ou [exécutons un script de module](#) , ce contexte d'exécution devient le sommet de la [pile de contextes d'exécution JavaScript](#) , au-dessus de laquelle un autre contexte d'exécution spécifique au script en question est poussé. (Cette configuration garantit que [ParseScript](#) et [l'évaluation](#) de [l'enregistrement du module de texte source](#) savent quel domaine utiliser.)

#### **Une carte de module**

Une [carte de module](#) qui est utilisée lors de l'importation de modules JavaScript.

#### **Un encodage de caractères d'URL d'API**

Encodage de caractères utilisé pour encoder les URL par des API appelées par des scripts qui utilisent cet [objet de paramètres d'environnement](#) .

#### **Une URL de base d'API**

Une [URL](#) utilisée par les API appelées par des scripts qui utilisent cet [objet de paramètres d'environnement](#) pour [analyser les URL](#) .

### **Une origine**

Une [origine](#) utilisée dans les contrôles de sécurité.

### **Un conteneur de politique**

Conteneur [de stratégies](#) contenant les stratégies utilisées pour les contrôles de sécurité.

### **Une capacité isolée cross-origin**

Un booléen indiquant si les scripts qui utilisent cet [objet de paramètres d'environnement](#) sont autorisés à utiliser les API qui nécessitent une isolation cross-origin.

### **Une origine temporelle**

Nombre utilisé comme référence pour les horodatages liés aux performances. [\[THS\]](#)

Un [objet de paramètres d'environnement](#) a également un **ensemble faible de promesses rejetées en attente** et une **liste de promesses rejetées sur le point d'être notifiées** , utilisées pour suivre [les rejets de promesses non gérées](#) . L' [ensemble faible des promesses rejetées en attente](#) ne doit pas créer de références fortes à l'un de ses membres, et les implémentations sont libres de limiter sa taille, par exemple en supprimant les anciennes entrées lorsque de nouvelles sont ajoutées.

La **boucle d'événement responsable** d'un [objet de paramètres d'environnement](#) est [la boucle d'événement](#) de l' [agent pertinent](#) de son [objet global](#) .

## **8.1.3.3 Domaines, objets de paramètres et objets globaux**

Un **objet global** est un objet JavaScript qui est le champ `[[GlobalObject]]` d'un [domaine](#) .

*Dans cette spécification, tous [les domaines](#) sont [créés](#) avec [des objets globaux](#) qui sont des objets `Window` ou `WorkerGlobalScope`.*

Il existe toujours un mappage 1-à-1-à-1 entre [les domaines](#) , [les objets globaux](#) et [les objets de paramètres d'environnement](#) :

- Un [domaine](#) a un champ `[[HostDefined]]`, qui contient l'**objet de paramètres du domaine** .
- Un [domaine](#) a un champ `[[GlobalObject]]`, qui contient l'**objet global du domaine** .

- Chaque [objet global](#) de cette spécification est créé lors de la [création](#) d'un [domaine](#) correspondant , appelé **domaine de l'objet global** .
- Chaque [objet global](#) dans cette spécification est créé à côté d'un [objet de paramètres d'environnement](#) correspondant , appelé son [objet de paramètres pertinent](#) .
- Le composant Realm du [contexte d'exécution du domaine d'un objet](#) de paramètres d'environnement est **le domaine de l'objet de paramètres d'environnement** .
- [Le domaine](#) d' un [objet de paramètres d'environnement](#) a alors un champ `[[GlobalObject]]`, qui contient **l'objet global de l'objet de paramètres d'environnement** .

Pour **créer un nouveau domaine** dans un [agent](#) *agent* , éventuellement avec des instructions pour créer un objet global ou une liaison **this globale (ou les deux)**, les **étapes suivantes sont suivies** :

1. Effectuez [InitializeHostDefinedRealm](#) () avec les personnalisations fournies pour créer l'objet global et la liaison globale **this** .
2. Soit *le contexte d'exécution du domaine* le [contexte d'exécution JavaScript en cours d'exécution](#) .

*Il s'agit du [contexte d'exécution JavaScript](#) créé à l'étape précédente.*

3. Supprimez *le contexte d'exécution du domaine* de la [pile de contexte d'exécution JavaScript](#) .
4. Soit *realm* le composant *Realm* du *contexte d'exécution* du realm.
5. Définissez l'agent du *domaine* sur *agent* . **Ce pointeur n'est pas encore défini dans la spécification JavaScript ; voir [tc39/ecma262#1357](#) .**
6. Si [le mode d'isolation cross-origin](#) du [cluster d'agents](#) de l' *agent* est " ", alors :[none](#)
  1. Soit *global* l' [objet global](#) du *domaine* .
  2. Que *le statut* soit ! *global* .`[[Supprimer]](" SharedArrayBuffer")`.
  3. [Assert](#) : *le statut* est vrai.

*Ceci est fait pour la compatibilité avec le contenu Web et il y a un espoir que cela puisse être supprimé à l'avenir. Les développeurs Web peuvent toujours accéder au constructeur via `new WebAssembly.Memory({ shared:true, initial:0, maximum:0 }).buffer.constructor`.*

7. Renvoie *le contexte d'exécution du domaine* .



---

Lors de la définition des étapes de l'algorithme tout au long de cette spécification, il est souvent important d'indiquer quel [domaine](#) doit être utilisé ou, de manière équivalente, quel [objet global](#) ou [objet de paramètres d'environnement](#) doit être utilisé. En général, il y a au moins quatre possibilités :

### **Entrée**

Cela correspond au script qui a lancé l'action de script en cours d'exécution : c'est-à-dire la fonction ou le script que l'agent utilisateur a appelé lorsqu'il a appelé le code de l'auteur.

### **Titulaire**

Cela correspond à la fonction ou au script d'auteur le plus récemment saisi sur la pile, ou à la fonction ou au script d'auteur qui a planifié à l'origine le rappel en cours d'exécution.

### **Actuel**

Cela correspond à l'objet de fonction en cours d'exécution, y compris les fonctions d'agent utilisateur intégrées qui peuvent ne pas être implémentées en tant que JavaScript. (Il est dérivé du [domaine actuel](#) .)

### **Pertinent**

Chaque [objet de plate-forme](#) a un [domaine pertinent](#) , qui est à peu près le [domaine](#) dans lequel il a été créé. [Lors de l'écriture d'algorithmes, l' objet de plate-forme](#) le plus important dont [le domaine pertinent](#) peut être important est la valeur **this** de l'objet de fonction en cours d'exécution. Dans certains cas, il peut y avoir d'autres [domaines pertinents](#) importants , tels que ceux de n'importe quel argument.

Notez que les concepts [d'entrée](#) , [de titulaire](#) et [d'actuel](#) sont utilisables sans réserve, alors que le concept [pertinent](#) doit être appliqué à un [objet de plate-forme](#) particulier .

***Les concepts [de titulaire](#) et [d'entrée](#) ne doivent pas être utilisés par de nouvelles spécifications, car ils sont excessivement compliqués et peu intuitifs à utiliser. Nous nous efforçons de supprimer presque toutes les utilisations existantes de la plateforme : voir [le numéro 1430](#) pour [l'opérateur historique](#) et [le numéro 1431](#) pour [l'entrée](#) .***

En général, les spécifications de la plate-forme Web doivent utiliser le concept [pertinent](#) , appliqué à l'objet sur lequel l'opération est effectuée (généralement la valeur **this** de la méthode actuelle). Cela ne correspond pas à la spécification JavaScript, où [current](#) est généralement utilisé par défaut (par exemple, pour déterminer le [domaine](#) dont `Array` le constructeur doit être utilisé pour construire le résultat dans `Array.prototype.map`). Mais cette incohérence est tellement ancrée dans la plate-forme que nous devons l'accepter à l'avenir.



Considérez les pages suivantes, `a.html` chargées dans une fenêtre de navigateur, `b.html` chargées dans un `iframe` comme indiqué, et `c.html` omises `d.html` (elles peuvent simplement être des documents vides) :

```
<!-- a.html -->
<!DOCTYPE html>
<html lang="en">
<title>Entry page</title>

<iframe src="b.html"></iframe>
<button onclick="frames[0].hello()">Hello</button>

<!--b.html -->
<!DOCTYPE html>
<html lang="en">
<title>Incumbent page</title>

<iframe src="c.html" id="c"></iframe>
<iframe src="d.html" id="d"></iframe>

<script>
  const c = document.querySelector("#c").contentWindow;
  const d = document.querySelector("#d").contentWindow;

  window.hello = () => {
    c.print.call(d);
  };
</script>
```

Chaque page a son propre [contexte de navigation](#) , et donc son propre [domaine](#) , [objet global](#) et [objet de paramètres d'environnement](#) .

Lorsque la `print()` méthode est appelée en réponse à l'appui sur le bouton dans `a.html`, alors :

- Le [domaine d'entrée](#) est celui de `a.html`.
- Le [domaine en place](#) est celui de `b.html`.
- Le [domaine actuel](#) est celui de `c.html` (puisque'il s'agit de la `print()` méthode à partir de `c.html` laquelle le code s'exécute).
- Le [domaine pertinent](#) de l'objet sur lequel la `print()` méthode est appelée est celui de `d.html`.

L'une des raisons pour lesquelles le concept [pertinent](#) est généralement un meilleur choix par défaut que le concept [actuel](#) est qu'il est plus adapté à la création d'un objet qui doit être persistant et renvoyé plusieurs fois. Par exemple, la `navigator.getBattery()` méthode crée des promesses dans le [domaine pertinent](#) pour l' [Navigator](#) objet sur lequel elle est invoquée. Cela a l'impact suivant : [\[BATTERIE\]](#)

```
<!-- outer.html -->
<!DOCTYPE html>
<html lang="en">
<title>Relevant realm demo: outer page</title>
<script>
  function doTest() {
    const promise = navigator.getBattery.call(frames[0].navigator);

    console.log(promise instanceof Promise);           // logs
false
    console.log(promise instanceof frames[0].Promise); // logs true

    frames[0].hello();
  }
</script>
<iframe src="inner.html" onload="doTest()"></iframe>

<!-- inner.html -->
<!DOCTYPE html>
<html lang="en">
<title>Relevant realm demo: inner page</title>
<script>
  function hello() {
    const promise = navigator.getBattery();

    console.log(promise instanceof Promise);           // logs true
    console.log(promise instanceof parent.Promise);    // logs false
  }
</script>
```

Si l'algorithme de la `getBattery()` méthode avait utilisé à la place le [domaine actuel](#) , tous les résultats seraient inversés. C'est-à-dire qu'après le premier appel à `getBattery()` in `outer.html`, l' [Navigator](#) objet in `inner.html` stockerait en permanence un `Promise` objet créé dans `outer.html` le [domaine](#) de , et des appels comme celui-ci à l'intérieur de la `hello()` fonction renverraient donc une promesse du "mauvais" domaine. Comme cela n'est pas souhaitable, l'algorithme utilise à la place

le [domaine pertinent](#) , donnant les résultats sensés indiqués dans les commentaires ci-dessus.

---

Le reste de cette section traite de la définition formelle des concepts [d'entrée](#) , [de titulaire](#) , [d'actuel](#) et [de pertinence](#) .

#### **8.1.3.3.1 Saisie**

Le processus d' [appel de scripts](#) va pousser ou faire apparaître [des contextes d'exécution de domaine](#) sur la [pile de contextes d'exécution JavaScript](#) , entrecoupés d'autres [contextes d'exécution](#) .

Avec cela en main, nous définissons le **contexte d'exécution d'entrée** comme étant l'élément le plus récemment poussé dans la [pile de contexte d'exécution JavaScript](#) qui est un [contexte d'exécution de domaine](#) . Le **domaine d'entrée** est le composant Domaine du [contexte d'exécution d'entrée](#) .

Ensuite, l' **objet de paramètres d'entrée** est l' [objet de paramètres d'environnement](#) du [domaine d'entrée](#) .

De même, l' **objet global d'entrée** est l' [objet global](#) du [domaine d'entrée](#) .

#### **8.1.3.3.2 Titulaire**

Tous [les contextes d'exécution JavaScript](#) doivent contenir, dans le cadre de leur état d'évaluation de code, une valeur **de compteur skip-when-determining-incumbent** , qui est initialement zéro. Lors du processus de [préparation de l'exécution d'un rappel](#) et [de nettoyage après l'exécution d'un rappel](#) , cette valeur sera incrémentée et décrétementée.

Chaque [boucle d'événements](#) est associée à une **pile d'objets de paramètres d'opérateur historique de sauvegarde** , initialement vide. En gros, il est utilisé pour déterminer l' [objet de paramètres en place](#) lorsqu'aucun code d'auteur n'est sur la pile, mais que le code d'auteur est responsable de l'exécution de l'algorithme actuel d'une manière ou d'une autre. Le processus de [préparation à l'exécution d'un rappel](#) et [de nettoyage après l'exécution d'un rappel](#) manipule cette pile. [\[WEBIDL\]](#)

Lorsque Web IDL est utilisé pour [invoquer](#) le code de l'auteur, ou lorsque [HostEnqueuePromiseJob](#) invoque un travail de promesse, ils utilisent les

algorithmes suivants pour suivre les données pertinentes pour déterminer l' [objet de paramètres en place](#) :

Pour **préparer l'exécution d'un rappel** avec un [objet de paramètres d'environnement](#) *settings* :

1. Transférez *les paramètres* vers la [pile d'objets de sauvegarde des paramètres en place](#) .
2. Soit *context* le [contexte d'exécution contenant le script le plus élevé](#) .
3. Si *le contexte* n'est pas nul, incrémente le [compteur skip-when-determining-incumbent](#) du contexte .

Pour **nettoyer après avoir exécuté un rappel** avec un [objet de paramètres d'environnement](#) *settings* :

1. Soit *context* le [contexte d'exécution contenant le script le plus élevé](#) .  
*Ce sera le même que le [contexte d'exécution de script le plus élevé](#) à l'intérieur de l'invocation correspondante de [prepare to run a callback](#) .*
2. Si *le contexte* n'est pas nul, décrémente le compteur [skip-when-determining-incumbent du contexte](#) .
3. [Assert](#) : l'entrée la plus haute de la [pile d'objets de paramètres de sauvegarde titulaire](#) est *settings* .
4. Supprimez *les paramètres* de la [pile d'objets de sauvegarde des paramètres en place](#) .

Ici, le **contexte d'exécution contenant le script le plus élevé** est l'entrée la plus élevée de la [pile de contexte d'exécution JavaScript](#) qui a un composant `ScriptOrModule` non nul, ou null s'il n'y a pas une telle entrée dans la [pile de contexte d'exécution JavaScript](#) .

Avec tout cela en place, l' **objet de paramètres titulaire** est déterminé comme suit :

1. Soit *context* le [contexte d'exécution contenant le script le plus élevé](#) .
2. Si *context* est nul, ou si le [compteur skip-when-determining-incumbent de context](#) est supérieur à zéro, alors :
  1. [Assert](#) : la [pile d'objets de sauvegarde des paramètres titulaires](#) n'est pas vide.

*Cette assertion échouerait si vous essayez d'obtenir l' [objet de paramètres en place](#) à l'intérieur d'un algorithme qui n'a été déclenché ni par l'[appel de scripts](#) ni par Web IDL [appelant](#) un rappel. Par exemple, il se déclencherait si vous tentiez d'obtenir l' [objet de paramètres titulaire](#) à l'intérieur d'un algorithme qui s'exécutait*

*périodiquement dans le cadre de la [boucle d'événements](#) , sans implication du code de l'auteur. Dans de tels cas, le concept [d'opérateur historique](#) ne peut pas être utilisé.*

2. Renvoie l'entrée la plus haute de la [pile d'objets des paramètres de l'opérateur historique de sauvegarde](#) .

3. Renvoie [l'objet de paramètres](#) du composant Realm du *contexte* .

Ensuite, le **domaine titulaire** est le [domaine](#) de l' [objet de paramètres titulaire](#) .

De même, l' **objet global titulaire** est l' [objet global](#) de l' [objet paramètres titulaire](#) .

---

La série d'exemples suivante vise à faire comprendre comment tous les différents mécanismes contribuent à la définition du concept [d'opérateur historique](#) :

Prenons l'exemple de démarrage suivant :

```
<!DOCTYPE html>
<iframe></iframe>
<script>
  frames[0].postMessage("some data", "*");
</script>
```

Il y a deux [objets de paramètres d'environnement](#) intéressants ici : celui de `window` et celui de `frames[0]`. Notre préoccupation est la suivante : quel est l' [objet de paramètres en place](#) au moment où l'algorithme s'exécute `postMessage()` ?

Ce devrait être celui de `window`, pour saisir la notion intuitive que le script de l'auteur responsable de l'exécution de l'algorithme s'exécute dans `window`, et non `frames[0]`. Cela a du sens : les [étapes de message de la fenêtre de publication](#) utilisent l' [objet de paramètres en place](#) pour déterminer la [source](#) propriété du résultat [MessageEvent](#), et dans ce cas, `window` il s'agit certainement de la source du message.

Expliquons maintenant comment les étapes indiquées ci-dessus nous donnent le résultat intuitivement souhaité de [l'objet de paramètres pertinent](#) `window` de .

Lorsque les [étapes de message de la fenêtre de publication](#) recherchent l' [objet de paramètres en place](#) , le [contexte d'exécution de script le plus élevé](#) sera celui correspondant à l' [script](#) élément : il a été poussé sur la [pile de contexte d'exécution JavaScript](#) dans le cadre de [ScriptEvaluation](#) lors de l' exécution d'un algorithme [de script classique](#) . Puisqu'il n'y a pas d'invocations de rappel Web IDL impliquées,

le [compteur skip-when-determining-incumbent](#) du contexte est égal à zéro, il est donc utilisé pour déterminer l' [objet de paramètres de titulaire](#) ; le résultat est l' [objet de paramètres d'environnement](#) de `window`.

(Notez comment l' [objet de paramètres d'environnement](#) de `frames[0]` est l' [objet de paramètres pertinent](#) de `this` au moment où la `postMessage()` méthode est appelée, et est donc impliqué dans la détermination de la *cible* du message. Alors que le titulaire est utilisé pour déterminer la *source* .)

Considérez l'exemple suivant plus compliqué :

```
<!DOCTYPE html>
<iframe></iframe>
<script>
    const bound = frames[0].postMessage.bind(frames[0], "some data",
    "*");
    window.setTimeout(bound);
</script>
```

Cet exemple est très similaire au précédent, mais avec une indirection supplémentaire via `Function.prototype.bind` ainsi que `setTimeout()`. Mais la réponse devrait être la même : invoquer des algorithmes de manière asynchrone ne devrait pas changer le concept [en place](#) .

Cette fois, le résultat implique des mécanismes plus compliqués :

Lorsque `bound` est [converti](#) en un type de rappel Web IDL, l' [objet de paramètres en place](#) est celui correspondant à `window` (de la même manière que dans notre exemple de démarrage ci-dessus). Web IDL stocke ceci en tant que [contexte de rappel](#) de la valeur de rappel résultante .

Lorsque la [tâche](#) publiée par `setTimeout()` s'exécute, l'algorithme de cette tâche utilise Web IDL pour [appeler](#) la valeur de rappel stockée. Web IDL appelle à son tour [la préparation ci-dessus pour exécuter un](#) algorithme de rappel. Cela pousse le [contexte de rappel](#) stocké sur la [pile d'objets des paramètres de l'opérateur historique de sauvegarde](#) . À ce moment (à l'intérieur de la tâche du minuteur), il n'y a pas de code d'auteur sur la pile, donc le [contexte d'exécution du script le plus élevé](#) est nul, et rien n'obtient son [compteur skip-when-determining-incumbent](#) incrémenté.

L'appel du rappel appelle alors `bound`, qui à son tour appelle la `postMessage()` méthode de `frames[0]`. Lorsque l' `postMessage()` algorithme recherche l' [objet de paramètres en place](#) , il n'y a toujours pas de code d'auteur sur la pile, car la fonction liée appelle directement la méthode intégrée. Ainsi, le [contexte d'exécution contenant le script le plus élevé](#) sera nul : la pile [de contextes d'exécution JavaScript](#) ne contient qu'un contexte d'exécution correspondant à `postMessage()`, sans contexte [ScriptEvaluation](#) ou similaire en dessous.

C'est là que nous retombons sur la [pile d'objets de sauvegarde des paramètres en place](#) . Comme indiqué ci-dessus, il contiendra comme entrée la plus élevée l' [objet de paramètres pertinent](#) de `window`. C'est donc ce qui est utilisé comme [objet de paramètres titulaire](#) lors de l'exécution de l' `postMessage()` algorithme.

Considérez cet exemple final, encore plus compliqué :

```
<!-- a.html -->
<!DOCTYPE html>
<button>click me</button>
<iframe></iframe>
<script>
const bound = frames[0].location.assign.bind(frames[0].location,
"https://example.com/");
document.querySelector("button").addEventListener("click", bound);
</script>
<!-- b.html -->
<!DOCTYPE html>
<iframe src="a.html"></iframe>
<script>
  const iframe = document.querySelector("iframe");
  iframe.onload = function onLoad() {
    iframe.contentWindow.document.querySelector("button").click();
  };
</script>
```

Encore une fois, il y a deux [objets de paramètres d'environnement](#) intéressants en jeu : celui de `a.html` et celui de `b.html`. Lorsque la `location.assign()` méthode déclenche l' algorithme [Location de navigation -objet](#) , quel sera l' [objet de paramètres en place](#) ? Comme précédemment, il devrait intuitivement s'agir de `a.html`: l' `click` auditeur était initialement programmé par `a.html`, donc même si quelque chose impliquant `b.html` provoque le renvoi de l'auditeur, le responsable [titulaire](#) est celui de `a.html`.

La configuration du rappel est similaire à l'exemple précédent : lorsque `bound` est [converti](#) en un type de rappel Web IDL, l' [objet de paramètres en place](#) est celui correspondant à `a.html`, qui est stocké en tant que [contexte de rappel](#) du rappel .

Lorsque la `click()` méthode est appelée à l'intérieur `b.html`, elle [distribue](#) un `click` événement sur le bouton qui se trouve à l'intérieur `a.html`. Cette fois, lorsque la [préparation de l'exécution d'un](#) algorithme de rappel s'exécute dans le cadre de la répartition des événements, il y a du code auteur sur la pile ; le [contexte d'exécution le plus élevé ayant un script](#) est celui de la `onLoad` fonction, dont [le compteur skip-when-determining-incumbent](#) est incrémenté. En outre, l'[objet de paramètres d'environnement](#) `a.html` de (stocké en

tant que [contexte de rappel de](#) ) est poussé sur la [pile d'objets de paramètres de sauvegarde titulaire](#) `.EventHandler`

Désormais, lorsque l' algorithme [Location de navigation -objet](#) recherche l' [objet de paramètres en place](#) , le [contexte d'exécution le plus élevé contenant le script](#) est toujours celui de la `onLoad` fonction (car nous utilisons une fonction liée comme rappel). Sa valeur [de compteur skip-when-determining-incumbent](#) est un, cependant, nous retombons à la [pile d'objets de sauvegarde des paramètres titulaires](#) . Cela nous donne l' [objet de paramètres d'environnement](#) de `a.html`, comme prévu.

Notez que cela signifie que même si c'est l' `iframe` intérieur `a.html` qui navigue, c'est `a.html` lui-même qui est utilisé comme source `Document`, qui détermine entre autres la [requête client](#) . C'est [peut-être la seule utilisation justifiable du concept d'opérateur historique sur la plateforme web](#) ; dans tous les autres cas, les conséquences de son utilisation sont tout simplement déroutantes et nous espérons un jour les faire passer à une utilisation [actuelle](#) ou [pertinente](#) , selon le cas.

#### 8.1.3.3 Courant

La spécification JavaScript définit le [domaine actuel](#) , également appelé "enregistrement de domaine actuel". [JAVASCRIPT](#)

Ensuite, l' **objet de paramètres actuel** est l' [objet de paramètres d'environnement](#) du [domaine actuel](#) .

De même, l' **objet global courant** est l' [objet global](#) du [domaine courant](#) .

#### 8.1.3.3.4 Pertinent

Le **domaine pertinent** pour un [objet de plate-forme](#) est la valeur de [son champ `\[\[Realm\]\]`](#) .

Ensuite, l' **objet de paramètres pertinent** pour un [objet de plate-forme](#) `o` est l' [objet de paramètres d'environnement](#) du [domaine pertinent](#) pour `o` .

De même, l' **objet global pertinent** pour un [objet plate-forme](#) `o` est l' [objet global](#) du [domaine pertinent](#) pour `o` .

#### 8.1.3.4 Activation et désactivation des scripts



**Le script est activé** pour *les paramètres* [d'un objet de paramètres d'environnement](#) lorsque toutes les conditions suivantes sont remplies :

- L'agent utilisateur prend en charge les scripts.
- L'utilisateur n'a pas désactivé les scripts pour *les paramètres* pour le moment. (Les agents utilisateurs peuvent offrir aux utilisateurs la possibilité de désactiver les scripts globalement ou de manière plus précise, par exemple, sur une base par origine, jusqu'au niveau des [objets de paramètres d'environnement](#) individuels .)
- Soit [l'objet global](#) de *settings* n'est pas un objet , soit [l'indicateur de sandboxing actif](#) de l' [objet global](#) de *settings* [associé](#) n'a pas son [indicateur de contexte de navigation des scripts sandbox](#) défini. [WindowDocument](#)

**Le script est désactivé** pour un [objet de paramètres d'environnement](#) lorsque le script n'est pas [activé](#) pour lui, c'est-à-dire lorsque l'une des conditions ci-dessus est fausse.

---

**Le script est activé** pour un *nœud* nœud si [le contexte de navigation](#) du [document de nœud](#) du *nœud* n'est pas nul, et [le script est activé](#) pour [l'objet de paramètres pertinent](#) du *nœud* .

**Le script est désactivé** pour un nœud lorsque le script n'est pas [activé](#) , c'est-à-dire lorsque [le contexte de navigation](#) de son [document de nœud](#) est nul ou lorsque [le script est désactivé](#) pour son [objet de paramètres pertinent](#) .

#### 8.1.3.5 Contextes sécurisés

Un *environnement* [environnement](#) est un **contexte sécurisé** si l'algorithme suivant renvoie true :

1. Si *environnement* est un [objet de paramètres d'environnement](#) , alors :
  1. Soit *global* l' [objet global](#) de *l'environnement* .
  2. Si *global* est un [WorkerGlobalScope](#), alors :

1. Si [l'objet de paramètres pertinent de l'ensemble propriétaire](#) de `global` [0] est un [contexte sécurisé](#) , alors renvoie `true`.

*Nous n'avons qu'à vérifier le 0ème élément puisqu'ils seront forcément tous cohérents.*

2. Renvoie faux.

3. Si `global` est un [WorkletGlobalScope](#), alors retourne `true`.

*Les worklets ne peuvent être créés que dans des contextes sécurisés.*

2. Si le résultat de [L'URL est-elle potentiellement digne de confiance ? l'URL de création de niveau supérieur](#) de `l'environnement` donné est " ", puis renvoie `true`. `Potentially Trustworthy`

3. Renvoie faux.

Un [environnement](#) est un **contexte non sécurisé** s'il ne s'agit pas d'un [contexte sécurisé](#) .

## 8.1.4 Modèle de traitement de script

### 8.1.4.1 Scénarios

Un **script** est l'une des trois [structures](#) possibles . Tous les scripts ont :

#### Un *objet paramètres*

Un [objet de paramètres d'environnement](#) , contenant divers paramètres partagés avec d'autres [scripts](#) dans le même contexte.

#### Un *enregistrement*

L'un des éléments suivants :

- un [enregistrement de script](#) , pour [les scripts classiques](#) ;
- un [enregistrement de module de texte source](#) , pour [les scripts de module JavaScript](#) ;
- un [enregistrement de module synthétique](#) , pour [les scripts de module CSS](#) et [les scripts de module JSON](#)
- `null`, représentant un échec d'analyse.

#### Une *erreur d'analyse*

Une valeur JavaScript, qui n'a de sens que si l' [enregistrement](#) est nul, indiquant que le texte source du script correspondant n'a pas pu être analysé.

*Cette valeur est utilisée pour le suivi interne des erreurs d'analyse immédiates lors [de la création de scripts](#) et ne doit pas être utilisée directement. Au lieu de cela, consultez l' [erreur à relancer](#) lors de la détermination de "ce qui n'a pas fonctionné" pour ce script.*

### Une erreur à relancer

Une valeur JavaScript représentant une erreur qui empêchera l'évaluation de réussir. Il sera relancé par toute tentative d' [exécution](#) du script.

*Il peut s'agir de l' [erreur d'analyse](#) du script , mais dans le cas d'un [script de module](#) , il peut s'agir de l' [erreur d'analyse](#) de l'une de ses dépendances ou d'une erreur de [résolution d'un spécificateur de module](#) . Étant donné que cette valeur d'exception est fournie par la spécification JavaScript, nous savons qu'elle n'est jamais nulle, nous utilisons donc null pour signaler qu'aucune erreur ne s'est produite.*

### Options de récupération

Un [script fetch options](#) , contenant diverses options liées à la récupération de ce script ou [des scripts de module](#) qu'il importe.

### Une URL de base

Une [URL](#) de base utilisée pour [résoudre les spécificateurs de module](#) . Il s'agira soit de l'URL à partir de laquelle le script a été obtenu, pour les scripts externes, soit de l' [URL de base](#) du document contenant le document, pour les scripts en ligne.

Un **script classique** est un type de [script](#) qui possède l' [élément](#) supplémentaire suivant :

### Un booléen d'erreurs en sourdine

Un booléen qui, s'il est vrai, signifie que les informations d'erreur ne seront pas fournies pour les erreurs dans ce script. Ceci est utilisé pour désactiver les erreurs pour les scripts d'origine croisée, car cela peut divulguer des informations privées.

Un **script de module** est un autre type de [script](#) . Il n'a pas [d'éléments](#) supplémentaires .

[Les scripts de module](#) peuvent être classés en trois types :

- Un [script de module](#) est un **script de module JavaScript** si son [enregistrement](#) est un [Source Text Module Record](#) .
- Un [script de module](#) est un **script de module CSS** si son [enregistrement](#) est un [enregistrement de module synthétique](#) et s'il a été créé via l' algorithme [de création d'un script de module CSS](#) . Les scripts de module CSS représentent une feuille de style CSS analysée.
- Un [script de module](#) est un **script de module JSON** si son [enregistrement](#) est un [enregistrement de module synthétique](#) et s'il a été créé via l' algorithme [de](#)

[création d'un script de module JSON](#) . Les scripts de module JSON représentent un document JSON analysé.

*Comme les feuilles de style CSS et les documents JSON n'importent pas de modules dépendants et ne génèrent pas d'exceptions lors de l'évaluation, les [options de récupération](#) et l'[URL de base](#) des [scripts de module CSS](#) et [des scripts de module JSON](#) sont toujours nulles.*

Le **script actif** est déterminé par l'algorithme suivant :

1. Soit *record* soit [GetActiveScriptOrModule](#) ().
2. Si l'enregistrement est nul, renvoie nul.
3. Renvoie l'enregistrement `.[[HostDefined]]`.

*Le concept [de script actif](#) n'est jusqu'à présent utilisé que par la [import\(\)](#) fonctionnalité, pour déterminer l' [URL de base](#) à utiliser pour résoudre les spécificateurs de module relatifs.*

#### 8.1.4.2 Récupérer des scripts

Cette section présente un certain nombre d'algorithmes pour récupérer des scripts, prenant diverses entrées nécessaires et aboutissant à [des scripts classiques](#) ou de module .

---

Les **options de récupération de script** sont une [structure](#) avec les [éléments](#) suivants :

##### ***nonce cryptographique***

Les [métadonnées nonce cryptographiques](#) utilisées pour la récupération initiale et pour récupérer tous les modules importés

##### ***métadonnées d'intégrité***

Les [métadonnées d'intégrité](#) utilisées pour la récupération initiale

##### ***métadonnées de l'analyseur***

Les [métadonnées de l'analyseur](#) utilisées pour la récupération initiale et pour récupérer tous les modules importés

##### ***mode d'identification***

Le [mode d'identification](#) utilisé pour la récupération initiale (pour [les scripts de module](#) ) et pour récupérer tous les modules importés (pour [les scripts de module](#) et [les scripts classiques](#) )

### **politique de référence**

La [politique de référence](#) utilisée pour la récupération initiale et pour récupérer tous les modules importés

### **blocage du rendu**

La valeur booléenne de [render-blocking](#) utilisée pour la récupération initiale et pour récupérer tous les modules importés. Sauf indication contraire, sa valeur est fausse.

### **récupérer la priorité**

La [priorité](#) utilisée pour la récupération initiale

*Rappelons que via la `import()` fonctionnalité, [les scripts classiques](#) peuvent importer [des scripts de module](#) .*

Les **options de récupération de script classiques par défaut** sont des [options de récupération de script](#) dont [le nonce cryptographique](#) est la chaîne vide, [les métadonnées d'intégrité](#) sont la chaîne vide, [les métadonnées de l'analyseur](#) sont "not-parser-inserted", [le mode d'identification](#) est "same-origin", [la stratégie de référence](#) est la chaîne vide et [la priorité de récupération](#) est "auto".

Étant donné une requête [request](#) et un [script fetch options](#) *options* , on définit :

### **configurer la demande de script classique**

Définissez les [métadonnées de nonce cryptographique de la requête](#) sur les [nonce cryptographiques](#) d' *options* , ses [métadonnées d'intégrité](#) sur les [métadonnées d'intégrité](#) des *options* , ses [métadonnées d'analyseur](#) sur les [métadonnées d' analyseur d' options](#) , sa [politique de référence](#) sur [la politique de référence](#) d' *options* , son [blocage de rendu](#) sur les *options* 's [render-blocking](#) , et sa [priorité](#) à [la priorité d'extraction](#) des *options* .

### **configurer la demande de script de module**

Définissez les [métadonnées nonce cryptographiques de la requête](#) sur les [nonce cryptographiques](#) des *options* , ses [métadonnées d' intégrité](#) sur les [métadonnées d' intégrité](#) des *options* , ses [métadonnées d' analyseur](#) sur les [métadonnées d' analyseur des options](#) , son [mode d' identification](#) sur [le mode d' identification](#) des *options* , sa [politique de référence](#) sur les *options* ' s [referrer policy](#) , son [blocage de rendu](#) vers le [blocage de rendu](#) des *options* , et sa [priorité](#) à [la priorité de récupération](#) des *options* .

Pour toute *option* [d'extraction de script](#) donnée , les **options d'extraction de script descendantes** sont de nouvelles [options d'extraction de script](#) dont [les éléments](#) ont

tous les mêmes valeurs, à l'exception des [métadonnées d'intégrité](#) , qui sont à la place la chaîne vide, et de la [priorité d'extraction](#) , qui est à la place " " .auto

---

Plusieurs des algorithmes ci-dessous peuvent être personnalisés avec un algorithme **d'exécution de l'extraction du hook** , qui prend une [requête](#) , un [isTopLevel](#) booléen et un algorithme **processCustomFetchResponse** . Il exécute [processCustomFetchResponse](#) avec une [réponse](#) et soit null (en cas d'échec), soit une [séquence d'octets](#) contenant le corps de la réponse. **isTopLevel** sera vrai pour toutes les extractions [de script classiques](#) , et pour l'extraction initiale lors [de l'extraction d'un graphique de script de module externe](#) ou [de l'extraction d'un graphique de script de travail de module](#) , mais faux pour les extractions résultant de `import` énoncés rencontrés tout au long du graphe ou à partir `import()` d'expressions.

Par défaut, si vous ne fournissez pas de [crochet perform the fetch, les algorithmes ci-dessous récupéreront](#) simplement la [requête](#) donnée , avec des personnalisations spécifiques à l'algorithme pour la [requête](#) et des validations de la [réponse](#) résultante .

Pour superposer vos propres personnalisations au-dessus de celles spécifiques à l'algorithme, fournissez un [crochet de récupération](#) qui modifie la [requête](#) donnée , la [récupère](#) , puis effectue des validations spécifiques de la [réponse](#) résultante (se terminant par une [erreur réseau](#) si les validations échouent).

Le crochet peut également être utilisé pour effectuer des personnalisations plus subtiles, telles que conserver un cache des [réponses](#) et éviter d'effectuer une [récupération](#) .

*Service Workers est un exemple de spécification qui exécute ces algorithmes avec ses propres options pour le crochet. [\[SW\]](#)*

---

Passons maintenant aux algorithmes eux-mêmes.

Pour **récupérer un script classique** à partir d'une *URL* , d'un *objet de paramètres* , de certaines *options* , d'un *paramètre CORS* , d'un *codage de caractères* et d'un algorithme *onComplete* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script classique](#) (en cas de succès).

1. Soit *request* le résultat de [la création d'une requête CORS potentielle](#) donnée *url* , " *script* " et *CORS setting* .

2. Définissez le [client](#) de la requête sur l'objet paramètres .
3. Définissez [le type d'initiateur](#) de la requête sur ".script
4. [Configurez la demande de script classique](#) en fonction de la demande et des options .
5. [Récupérer](#) la requête avec les étapes [processResponseConsumeBody](#) suivantes en fonction de la réponse [réponse](#) et null, échec ou une [séquence d'octets](#) bodyBytes :

*La réponse peut être [CORS-same-origin](#) ou [CORS-cross-origin](#) . Cela n'affecte que la façon dont le rapport d'erreurs se produit.*

1. Définissez la réponse sur la réponse [non sécurisée de la réponse](#) .
2. Si l'une des conditions suivantes est remplie :
  - bodyBytes est null ou échec ; ou
  - [le statut](#) de la réponse n'est pas un [statut ok](#) ,puis exécutez onComplete étant donné null et abandonnez ces étapes.

*Pour des raisons historiques, cet algorithme n'inclut pas la vérification de type MIME, contrairement aux autres algorithmes de récupération de scripts de cette section.*

3. Soit potentialMIMETypeForEncoding le résultat de [l'extraction d'un type MIME](#) donné de la liste d' [en-tête de la](#) réponse .
4. Définissez l'encodage des caractères sur le résultat de [l'extraction héritée d'un encodage](#) donné potentielMIMETypeForEncoding et encodage des caractères .

*Cela ignore intentionnellement l' [essence du type MIME](#) .*

5. Laissez le texte source être le résultat du [décodage](#) de bodyBytes en Unicode, en utilisant le codage de caractères comme codage de secours.

*L' algorithme [de décodage](#) remplace le codage des caractères si le fichier contient une nomenclature.*

6. Laissez les erreurs muettes être vraies si la réponse était [CORS-cross-origin](#) , et fausses sinon.
7. Soit script le résultat de [la création d'un script classique](#) avec le texte source , l'objet de paramètres , [l'URL](#) de la réponse , les options et les erreurs masquées .
8. Exécuter onComplete script donné .



Pour **récupérer un script de travail classique** en fonction d'une *url* , d'un *objet de paramètres de client de récupération* , d'une *destination* , d'un *objet de paramètres de script* , d'un algorithme *onComplete* et d'une [exécution facultative du hook de récupération](#) *performFetch* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script classique](#) (en cas de succès).

1. Soit *request* une nouvelle [requête](#) dont [l'URL](#) est *url* , [le client](#) récupère *l'objet des paramètres du client* , [la destination](#) est *la destination* , [le type d'initiateur](#) est " *other*", [le mode](#) est " *same-origin*", [le mode d'identification](#) est " *same-origin*", [les métadonnées de l'analyseur](#) sont " *not parser-inserted*" et dont [l'URL d'utilisation L'indicateur -credentials](#) est défini.
2. Si *performFetch* a été donné, exécutez *performFetch* avec *request* , *true* et avec *processResponseConsumeBody* comme défini ci-dessous.

Sinon, [récupérez](#) *la requête* avec [processResponseConsumeBody](#) défini sur *processResponseConsumeBody* comme défini ci-dessous.

Dans les deux cas, laissez *processResponseConsumeBody* donner *la réponse* [response](#) et null, échec ou une [séquence d'octets](#) *bodyBytes* être l'algorithme suivant :

1. Définissez *la réponse* sur la réponse [non sécurisée de la réponse](#) .
2. Si l'une des conditions suivantes est remplie :
  - *bodyBytes* est null ou échec ; ou
  - [le statut](#) de *la réponse* n'est pas un [statut ok](#) ,puis exécutez *onComplete* étant donné null et abandonnez ces étapes.
3. Si les deux conditions suivantes sont remplies :
  - [le schéma](#) de l' [URL](#) de *la réponse* est un [schéma HTTP\(S\)](#) ; et
  - le résultat de [l'extraction d'un type MIME](#) de [la liste](#) d'en-tête de *la réponse* n'est pas un [type JavaScript MIME](#) ,

puis exécutez *onComplete* étant donné null et abandonnez ces étapes.

*Les autres [schémas de récupération](#) sont exemptés de la vérification du type MIME pour des raisons de compatibilité Web historique. Nous pourrions peut-être resserrer cela à l'avenir; voir [numéro 3255](#) .*

4. Laissez *le texte source* être le résultat du [décodage UTF-8](#) *bodyBytes* .
5. Soit *script* le résultat de [la création d'un script classique](#) à l'aide *du texte source* , de *l'objet de paramètres de script* , de [l'URL](#) de *la réponse* et des [options de récupération de script classique par défaut](#) .
6. Exécuter *onComplete* *script* donné .



Pour **récupérer un script classique importé par le nœud de calcul** en fonction d'une *URL* , d'un *objet de paramètres* et d'un [hook de récupération facultatif](#) *performFetch* , exécutez ces étapes. L'algorithme se terminera de manière synchrone avec un [script classique](#) en cas de succès ou lancera une exception en cas d'échec.

1. Soit *la réponse* nulle.
2. Laissez *bodyBytes* être nul.
3. Soit *request* une nouvelle [requête](#) dont [l'URL](#) est *url* , [le client](#) est *l'objet settings* , [la destination](#) est " *script* " , [le type d'initiateur](#) est " *other* " , [les métadonnées de l'analyseur](#) sont " *not parser-inserted* " et dont [l'indicateur use-URL-credentials](#) est défini.
4. Si *performFetch* a été donné, exécutez *performFetch* avec *request* , *isTopLevel* et avec *processResponseConsumeBody* comme défini ci-dessous.

Sinon, [récupérez](#) la requête avec [processResponseConsumeBody](#) défini sur *processResponseConsumeBody* comme défini ci-dessous.

Dans les deux cas, supposons que *processResponseConsumeBody* donne [la réponse](#) *res* et null, échec ou une [séquence d'octets](#) *bb* soit l'algorithme suivant :

1. Définissez *bodyBytes* sur *bb* .
  2. Définissez *la réponse* sur *res* .
5. [Pause](#) jusqu'à ce que *la réponse* ne soit pas nulle.

*Contrairement aux autres algorithmes de cette section, le processus de récupération est ici synchrone.*

6. Définissez *la réponse* sur la réponse [non sécurisée de la réponse](#) .
7. Si l'une des conditions suivantes est remplie :
  1. *bodyBytes* est null ou échec ;
  2. [le statut](#) de *la réponse* n'est pas un [statut ok](#) ; ou
  3. le résultat de [l'extraction d'un type MIME](#) de [la liste](#) d'en-tête de *la réponse* n'est pas un [type JavaScript MIME](#) ,puis jetez un " [NetworkError](#) " [DOMException](#) .
8. Laissez *le texte source* être le résultat du [décodage UTF-8](#) *bodyBytes* .
9. Laissez *les erreurs muettes* être vraies si *la réponse* était [CORS-cross-origin](#) , et fausses sinon.

10. Soit *script* le résultat de [la création d'un script classique](#) avec le *texte source* , l'*objet de paramètres* , l'*URL* de la *réponse* , les [options de récupération de script classique par défaut](#) et les *erreurs masquées* .

11. *Scénario* de retour .

Pour **récupérer un graphique de script de module externe** en fonction d'une *url* , d'un *objet de paramètres* , de certaines *options* et d'un algorithme *onComplete* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script de module](#) (en cas de succès).

1. [Interdire l'importation ultérieure de cartes](#) en fonction de l'*objet de paramètres* .
2. [Récupérez un seul script de module](#) donné *url* , *settings object* , "*script*" , *options* , *settings object* , "*client*" , true, et avec les étapes suivantes, le *résultat* est donné :
  1. Si le *résultat* est nul, exécutez *onComplete* étant donné la valeur nulle et abandonnez ces étapes.
  2. [Récupère les descendants de et](#) le *résultat* du lien étant donné l'*objet de paramètres* , "*script*" et *onComplete* .

Pour **récupérer un graphique de script de module de préchargement de module** en fonction d'une *url* , d'une *destination* , d'un *objet de paramètres* , de certaines *options* et d'un algorithme *onComplete* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script de module](#) (en cas de succès).

1. [Interdire l'importation ultérieure de cartes](#) en fonction de l'*objet de paramètres* .
2. [Récupérez un seul script de module](#) donné *url* , *settings object* , *destination* , *options* , *settings object* , "*client*" , true, et avec les étapes suivantes, le *résultat* est donné :
  1. Run *onComplete* *résultat* donné .
  2. Si le *résultat* n'est pas nul, [récupérez éventuellement les descendants de et liez](#) le *résultat* en fonction des *paramètres objet* , *destination* et un algorithme vide.

Généralement, l'exécution de cette étape sera bénéfique pour les performances, car elle permet de précharger les modules qui seront invariablement demandés ultérieurement, via des algorithmes tels que la récupération d' [un graphe de script de module externe](#) qui récupère l'intégralité du graphe. Cependant, les agents utilisateurs pourraient souhaiter les ignorer dans des situations où la bande passante est limitée, ou dans des situations où les extractions pertinentes sont déjà en cours.

Pour **récupérer un graphique de script de module en ligne** à partir d'un *texte source* , d'une *URL de base* , d'un *objet de paramètres* , d'*options* et d'un algorithme *onComplete* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script de module](#) (en cas de succès).

1. [Interdire l'importation ultérieure de cartes](#) en fonction de l'*objet de paramètres* .
2. Soit *script* le résultat de [la création d'un script de module JavaScript](#) à l'aide du *texte source* , de l'*objet de paramètres* , de l'*URL de base* et des *options* .
3. Si le *script* est nul, exécutez *onComplete* avec la valeur null et retournez.
4. [Récupère les descendants d'un](#) *script* de lien , étant donné l'*objet de paramètres* , " *script* " et *onComplete* .

Pour **récupérer un graphique de script de travail de module** donné une *url* , un *objet de paramètres de client de récupération* , une *destination* , un *mode d'identification* , un *objet de paramètres de carte de module* et un algorithme *onComplete* , [récupérez un graphique de script de travail de worklet/module](#) donné *url* , récupérez l'*objet de paramètres de client* , *destination* , *mode d'identification* , *objet de paramètres de carte de module* et *onComplete* .

Pour **récupérer un graphique de script de worklet** donné une *url* , un *objet de paramètres de client de récupération* , une *destination* , un *mode d'informations d'identification* , un *objet de paramètres de carte de module* , un *moduleResponsesMap* et un algorithme *onComplete* , [récupérez un graphique de script de travail de worklet/module](#) donné *url* , récupérez le *client objet de paramètres* , *destination* , *mode d'informations d'identification* , *objet de paramètres de carte de module* , *onComplete* et les éléments suivants [effectuent la](#) requête d'extraction donnée et [processCustomFetchResponse](#) :

1. Soit *requestURL* l' [URL](#) de la requête .
2. Si *moduleResponsesMap* [ *requestURL* ] est " *fetching* ", attendez [en parallèle](#) jusqu'à ce que la valeur de cette entrée change, puis [mettez une tâche en file d'attente](#) sur la [source de tâche réseau](#) pour continuer à exécuter les étapes suivantes.
3. Si *moduleResponsesMap* [ *requestURL* ] [existe](#) , alors :
  1. Soit mis en *cache moduleResponsesMap* [ *requestURL* ] .
  2. Exécutez *processCustomFetchResponse* avec *mis en cache* [0] et *mis en cache* [1].
  3. Retour.
4. [Définissez](#) *moduleResponsesMap* [ *requestURL* ] sur " *fetching* ".

5. [Fetch request](#) , avec [processResponseConsumeBody](#) défini sur les étapes suivantes en fonction de la réponse [réponse](#) et null, échec ou une [séquence d'octets](#) *bodyBytes* :
    1. Définissez *moduleResponsesMap [ requestURL ]* sur ( *response* , *bodyBytes* ).
    2. Exécutez *processCustomFetchResponse* avec *response* et *bodyBytes* .
- 

Les algorithmes suivants sont destinés à un usage interne par cette spécification uniquement dans le cadre de [la récupération d'un graphe de script de module externe](#) ou d'autres concepts similaires ci-dessus, et ne doivent pas être utilisés directement par d'autres spécifications.

Ce diagramme illustre comment ces algorithmes sont liés à ceux ci-dessus, ainsi qu'entre eux :

[fetch an external module script graphfetch a modulepreload module script graphfetch an inline module script graphfetch a module worker script graphfetch a worklet script graphfetch a worklet/module worker script graphfetch the descendants of and link a module script](#)

Pour **récupérer un graphique de script de travail de worklet/module** en fonction d'une *url* , d'un *objet de paramètres de client de récupération* , d'une *destination* , d'un *mode d'informations d'identification* , d'un *objet de paramètres de carte de module* , d'un algorithme *onComplete* et d'une [exécution facultative du hook de récupération](#) *performFetch* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script de module](#) (en cas de succès).

1. Soit *options* une [option de récupération de script](#) dont [le nonce cryptographique](#) est la chaîne vide, [les métadonnées d'intégrité](#) sont la chaîne vide, [les métadonnées de l'analyseur](#) sont " `not-parser-inserted`", [le mode d'identification](#) est le *mode d'identification* , [la politique de référence](#) est la chaîne vide et [la priorité de récupération](#) est " `auto`".
2. [Récupérez un seul script de module](#) en fonction de *l'url* , *récupérez l'objet des paramètres du client* , *la destination* , *les options* , *l'objet des paramètres de la carte du module* , " `client`", *true* et *onSingleFetchComplete* comme défini ci-dessous. Si *performFetch* a été donné, transmettez-le également.

*onSingleFetchComplete* le résultat donné est l'algorithme suivant :

1. Si le *résultat* est nul, exécutez *onComplete* étant donné la valeur nulle et abandonnez ces étapes.

2. [Récupérer les descendants de et](#) le résultat du lien en fonction de l'objet , de la destination et de onComplete des paramètres du client de récupération . Si performFetch a été donné, transmettez-le également.

Pour **récupérer les descendants et lier un script de module** script de module , étant donné un *objet de paramètres client de récupération* , une *destination* , un algorithme *onComplete* et une option [d'exécution du hook de récupération](#) *performFetch* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script de module](#) (en cas de succès).

1. Soit record [l'enregistrement](#) du script du module .
2. Si *l'enregistrement* est nul, alors :
  1. Définissez l' erreur du script du module [pour renvoyer l' erreur d'analyse](#) du script du module .
  2. Exécutez *onComplete* script de module donné .
  3. Retour.
3. Soit *state* soit [Record](#) { [[ParseError]] : null, [[Destination]] : *destination* , [[PerformFetch]] : null }.
4. Si *performFetch* a été donné, définissez l'état .[[PerformFetch]] sur *performFetch* .
5. Laissez *loadingPromise* être record . [LoadRequestedModules](#) ( état ).

*Cette étape chargera de manière récursive toutes les dépendances transitives du module.*

6. [Une fois l'exécution](#) de *loadingPromise* effectuée , exécutez les étapes suivantes :

1. Effectuer *l'enregistrement* . [Lien](#) () .

*Cette étape appellera récursivement [Link](#) sur toutes les dépendances non liées du module.*

Si cela lève une exception, attrapez-la et définissez l'erreur du script de module [pour qu'elle renvoie](#) à cette exception.

2. Exécutez *onComplete* script de module donné .

7. [En cas de rejet](#) de *loadingPromise* , exécutez les étapes suivantes :

1. Si l'état .[[ParseError]] n'est pas nul, définissez [l'erreur](#) du script de module sur l'état .[[ParseError]] et exécutez *onComplete* le script de module donné .

2. Sinon, exécutez *onComplete* avec null.

*state .[[ParseError]] est nul lorsque loadingPromise est rejeté en raison d'une erreur de chargement.*

Pour **récupérer un seul script de module**, étant donné une *url*, un *objet de paramètres de client de récupération*, une *destination*, certaines *options*, un *objet de paramètres de carte de module*, un *réfèrent*, un *moduleRequest* facultatif, un booléen *isTopLevel*, un algorithme *onComplete* et un facultatif *effectuer la récupération hook performFetch*, exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un *script de module* (en cas de succès).

1. Laissez *moduleType* être " javascript".
2. Si *moduleRequest* a été donné, définissez *moduleType* sur le résultat de l'exécution du *type de module à partir* des étapes de demande de module *données moduleRequest*.
3. *Assert* : le résultat de l'exécution des étapes *autorisées du type de module* donné *moduleType* et l'*objet de paramètres de carte de module* est vrai. Sinon, nous n'aurions pas atteint ce point car un échec aurait été signalé lors de l'inspection de *moduleRequest* .[[Assertions]] lors de *la création d'un script de module JavaScript* ou *de la récupération d'un seul script de module importé*.
4. Soit *moduleMap* la *carte de modules* de l'*objet de paramètres* de *carte de module*.
5. Si *moduleMap* [( *url*, *moduleType* )] est " fetching", attendez *en parallèle* jusqu'à ce que la valeur de cette entrée change, puis *mettez une tâche en file d'attente* sur la *source de tâche réseau* pour continuer à exécuter les étapes suivantes.
6. Si *moduleMap* [( *url*, *moduleType* )] *existe*, exécutez *onComplete moduleMap* donné [( *url*, *moduleType* )] et revenez.
7. *Définissez moduleMap* [( *url*, *moduleType* )] sur " fetching".
8. Soit *request* une nouvelle *requête* dont l'*URL* est *url*, *la destination* est la *destination*, *le mode* est " cors", *le réfèrent* est le *réfèrent* et *le client* récupère l'*objet des paramètres du client*.
9. Si *la destination* est " worker", " sharedworker" ou " serviceworker" et que l'indicateur *de récupération du module de niveau supérieur* est défini, définissez *le mode* de la *demande* sur " .same-origin
10. Définissez *le type d'initiateur* de la *requête* sur " .script
11. *Configurez la demande de script de module* en fonction de la *demande* et des *options*.



12. Si *performFetch* a été donné,  
exécutez *performFetch* avec *request* , *isTopLevel* et  
avec *processResponseConsumeBody* comme défini ci-dessous.

Sinon, récupérez la requête avec *processResponseConsumeBody* défini  
sur *processResponseConsumeBody* comme défini ci-dessous.

Dans les deux cas, laissez *processResponseConsumeBody* donner la  
réponse response et null, échec ou une séquence d'octets *bodyBytes* être  
l'algorithme suivant :

La réponse est toujours CORS-same-origin .

1. Si l'une des conditions suivantes est remplie :
  - *bodyBytes* est null ou échec ; ou
  - le statut de la réponse n'est pas un statut ok ,puis définissez *moduleMap* [( *url* , *moduleType* )] sur null,  
exécutez *onComplete* étant donné null et abandonnez ces étapes.
2. Laissez le texte source être le résultat du décodage UTF-8 *bodyBytes* .
3. Soit le *type MIME* le résultat de l'extraction d'un type MIME de la liste d'en-tête de la réponse .
4. Laissez le script du module être nul.
5. Si le *type MIME* est un type JavaScript MIME et *moduleType* est  
" *javascript* " , alors définissez *module script* sur le résultat de la création d'un script de module JavaScript en fonction du texte  
source , de l' objet des paramètres de mappage du module , de l'  
URL de la réponse et des options .
6. Si l' essence de type MIME du *type MIME* est " *text/css* " et  
que *moduleType* est " *css* " , définissez le script de module sur le résultat  
de la création d'un script de module CSS en fonction du texte  
source et de l'objet de paramètres de carte de module .
7. Si l'essence du *type MIME* est un type JSON MIME et que  
*moduleType* est " *json* " , définissez le script de module sur le résultat  
de la création d'un script de module JSON en fonction du texte  
source et de l'objet de paramètres de carte de module .
8. Définissez *moduleMap* [( *url* , *moduleType* )] sur *module script* , et  
exécutez *onComplete* given *module script* .

Il est intentionnel que la carte du module soit indexée par l' URL de la demande , alors que l' URL de base du script du module est définie sur l' URL de la réponse . Le premier est utilisé pour dédupliquer les récupérations, tandis que le second est utilisé pour la résolution d'URL.

Pour **récupérer un seul script de module importé** , en fonction d'une *url* , d'un *objet de paramètres* , d'une *destination* , de certaines *options* , d'un *réfèrent* , d'un *moduleRequest* , d'un algorithme *onComplete* et d'une option [d'exécution du hook de récupération](#) *performFetch* , exécutez ces étapes. *onComplete* doit être un algorithme acceptant null (en cas d'échec) ou un [script de module](#) (en cas de succès).

1. [Assert](#) : *moduleRequest* .[[Assertions]] ne contient aucune *entrée Record* telle que *l'entrée* .[[Key]] n'est pas " " , car nous n'avons demandé que des assertions " " dans [HostGetSupportedImportAssertions](#) .*type**type*
2. Soit *moduleType* le résultat de l'exécution du [type de module à partir](#) des étapes de requête de *module données moduleRequest* .
3. Si le résultat de l'exécution des étapes [autorisées du type de module étant donné](#) *moduleType* et *l'objet de paramètres* est faux, alors exécutez *onComplete* étant donné null, et retournez.
4. [Récupérez un seul script de module](#) avec *url* , *settings* *object* , *destination* , *options* , *settings object* , *referrer* , *moduleRequest* , *false* et *onComplete* . Si *performFetch* a été donné, transmettez-le également.

#### 8.1.4.3 Création de scripts

Pour **créer un script classique** , à partir d'une *source* [de chaîne](#) , d'un objet de *paramètres* [d'environnement](#) , d'une [URL](#) *baseURL* , de certaines *options* [d'extraction de script](#) et d'un booléen facultatif *mutedErrors* (faux par défaut) :

1. Si *mutedErrors* est vrai, définissez *baseURL* sur [about:blank](#).  

*Lorsque mutedErrors a la valeur true, baseURL est l' [URL](#) de la [réponse CORS-cross-origin](#) du script , qui ne doit pas être exposée à JavaScript. Par conséquent, baseURL est filtré ici.*
2. Si [le script est désactivé](#) pour *settings* , définissez *source* sur la chaîne vide.
3. Soit *script* un nouveau [script classique](#) que cet algorithme initialisera par la suite.
4. Définissez [l'objet de paramètres](#) du *script* sur *settings* .
5. Définissez [l'URL de base](#) du *script* sur *baseURL* .
6. Définissez [les options de récupération](#) du *script* sur *options* .
7. Définissez [les erreurs masquées](#) du *script* sur *mutedErrors* .



8. Définissez [l'erreur d'analyse](#) du *script* et [l'erreur à relancer](#) sur null.
9. Laissez le *résultat* être [ParseScript](#) ( *source* , [domaine](#) des *paramètres* , *script* ).

*Passer script comme dernier paramètre ici garantit que le résultat `.[[HostDefined]]` sera script .*

10. Si le *résultat* est une [liste](#) d'erreurs, alors :
  1. Définissez [l'erreur d'analyse](#) du *script* et son [erreur à renvoyer](#) au *résultat* [0].
  2. Scénario de retour .
11. Définissez [l'enregistrement](#) du *script* sur *result* .
12. Scénario de retour .

Pour **créer un script de module JavaScript** , étant donné une *source* [de chaîne](#) , des *paramètres* [d'objet de paramètres d'environnement](#) , une [URL](#) *baseURL* et certaines *options* [d'options de récupération de script](#) :

1. Si [le script est désactivé](#) pour *settings* , définissez *source* sur la chaîne vide.
2. Soit *script* un nouveau [script de module](#) que cet algorithme initialisera par la suite.
3. Définissez [l'objet de paramètres](#) du *script* sur *settings* .
4. Définissez [l'URL de base](#) du *script* sur *baseURL* .
5. Définissez [les options de récupération](#) du *script* sur *options* .
6. Définissez [l'erreur d'analyse](#) du *script* et [l'erreur à relancer](#) sur null.
7. Soit *result* être [ParseModule](#) ( *source* , [domaine](#) des *paramètres* , *script* ).

*Passer script comme dernier paramètre ici garantit que le résultat `.[[HostDefined]]` sera script .*

8. Si le *résultat* est une [liste](#) d'erreurs, alors :
  1. Définissez [l'erreur d'analyse](#) du *script* sur le *résultat* [0].
  2. Scénario de retour .
9. [Assert](#) : *required* `.[[Assertions]]` ne contient aucune *entrée* [Record](#) telle que l'*entrée* `.[[Key]]` n'est pas " " , car nous n'avons demandé que " " des assertions dans [HostGetSupportedImportAssertions](#) .`type`
10. [Pour chaque enregistrement](#) [ModuleRequest](#) demandé du *résultat* `.[[RequestedModules]]` :

1. Soit *url* le résultat de [la résolution d'un spécificateur de module](#) donné par *le script* et *demandé* .[[Spécifier]], interceptant toutes les exceptions.
2. Si l'étape précédente a généré une exception, alors :
  1. Définissez [l'erreur d'analyse](#) du *script* sur cette exception.
  2. *Scénario* de retour .
3. Laissez *moduleType* être le résultat de l'exécution du [type de module à partir](#) des étapes de demande de module *données request* .
4. Si le résultat de l'exécution des étapes [autorisées du type de module](#) donné *moduleType* et *paramètres* est faux, alors :
  1. Soit *erreur* une nouvelle `TypeError` exception.
  2. Définissez [l' erreur d' analyse](#) du *script* sur *error* .
  3. *Scénario* de retour .

*Cette étape consiste essentiellement à valider tous les spécificateurs de module demandés et les assertions de type. Nous traitons un module avec des spécificateurs de module insolubles ou des assertions de type non prises en charge de la même manière qu'un module qui ne peut pas être analysé ; dans les deux cas, un problème de syntaxe rend impossible d'envisager de lier le module ultérieurement.*

11. Définissez [l'enregistrement](#) du *script* sur *result* .
12. *Scénario* de retour .

Pour **créer un script de module CSS** , à partir d'une *source* de chaîne et d'un [objet de paramètres d' environnement](#) :

1. Soit *script* un nouveau [script de module](#) que cet algorithme initialisera par la suite.
2. Définissez [l'objet de paramètres](#) du *script* sur *settings* .
3. Définissez [l'URL de base](#) du *script* et [les options d'extraction](#) sur null.
4. Définissez [l'erreur d'analyse](#) du *script* et [l'erreur à relancer](#) sur null.
5. Soit *feuille* le résultat de l'exécution des étapes pour [créer un construit CSSStyleSheet](#) avec un dictionnaire vide comme argument.
6. Exécutez les étapes pour [remplacer de manière synchrone les règles d'une CSSStyleSheet](#) *source* donnée sur *la feuille* .

Si cela lève une exception, attrapez-la et définissez [l'erreur d'analyse](#) du *script* sur cette exception et renvoyez *script* .

*Les étapes pour [remplacer de manière synchrone les règles d'un CSSStyleSheet](#) lanceront si la source contient des `@import` règles. C'est une conception pour l'instant car il n'y a pas encore d'accord sur la façon de les gérer pour les scripts de module CSS ; par conséquent, ils sont complètement bloqués jusqu'à ce qu'un consensus soit atteint.*

7. Définissez [l'enregistrement](#) du *script* sur le résultat de [CreateDefaultExportSyntheticModule](#) ( *feuille* ).

8. Scénario de retour .

Pour **créer un script de module JSON** , à partir d'une *source* de chaîne et d'un [objet de paramètres d'](#) *environnement* :

1. Soit *script* un nouveau [script de module](#) que cet algorithme initialisera par la suite.
2. Définissez [l'objet de paramètres](#) du *script* sur *settings* .
3. Définissez [l'URL de base](#) du *script* et [les options d'extraction](#) sur null.
4. Définissez [l'erreur d'analyse](#) du *script* et [l'erreur à relancer](#) sur null.
5. Laissez le résultat être [ParseJSONModule](#) ( *source* ).

Si cela lève une exception, attrapez-la et définissez [l'erreur d'analyse](#) du *script* sur cette exception et renvoyez *script* .

6. Définissez [l'enregistrement](#) du *script* sur *result* .

7. Scénario de retour .

Le **type de module des** étapes de demande de module, étant donné un [ModuleRequest Record](#) *moduleRequest* , est le suivant :

1. Laissez *moduleType* être " javascript".
2. Si *moduleRequest* .[[Assertions]] a une *entrée* [Record](#) telle que l'*entrée* .[[Key]] est " ", alors :
  1. Si l'*entrée* .[[Valeur]] est " javascript", alors définissez *moduleType* sur null.

*Cette spécification utilise le javascript type de module " " en interne pour [les scripts de module JavaScript](#) , cette étape est donc nécessaire pour empêcher l'importation de modules à l'aide d'une javascript assertion de type " " (un *moduleType* nul entraînera l'échec de la vérification [autorisée du type de module](#) ).*

2. Sinon, définissez *moduleType* sur l'entrée `.[[Value]]`.
3. Renvoie le type de module .

Les étapes **autorisées pour le type de module** , étant donné une chaîne *moduleType* et un objet de paramètres d'environnement settings , sont les suivantes :

1. Si *moduleType* n'est pas " javascript", " css" ou " json", alors renvoie false.
2. Si *moduleType* est " css" et que l' CSSStyleSheet interface n'est pas exposée dans le domaine des *paramètres* , alors renvoyez false.
3. Renvoie vrai.

#### 8.1.4.4 Appel de scripts

Pour **exécuter un script classique** avec un *script* de script classique et une *erreur de relance* booléenne facultative (false par défaut) :

1. Soit *settings* l' objet settings de *script* .
2. Vérifiez si nous pouvons exécuter un script avec *settings* . Si cela renvoie "ne pas exécuter", renvoyez NormalCompletion (vide).
3. Préparez-vous à exécuter le script avec *les paramètres* .
4. Laissez *evaluationStatus* être nul.
5. Si l'erreur du *script à relancer* n'est pas nulle, définissez *evaluationStatus* sur Completion { `[[Type]]: throw`, `[[Value]]: script's error to rethrow`, `[[Target]]: empty` }.
6. Sinon, définissez *evaluationStatus* sur ScriptEvaluation ( enregistrement du *script* ).

Si ScriptEvaluation ne se termine pas parce que l'agent utilisateur a abandonné le script en cours d'exécution , laissez *evaluationStatus* sur null.

7. Si *evaluationStatus* est un achèvement brutal , alors :
  1. Si *les erreurs de relance* sont vraies et que les erreurs ignorées du *script* sont fausses, alors :
    1. Nettoyer après avoir exécuté le script avec *les paramètres* .
    2. Renvoyez *evaluationStatus* `.[[Valeur]]`.

2. Si *les erreurs de relance* sont vraies et que [les erreurs ignorées](#) du *script* sont vraies, alors :
  1. [Nettoyer après avoir exécuté le script](#) avec *les paramètres* .
  2. Lancez un "[NetworkError](#)" [DOMException](#) .
3. Sinon, *relancer les erreurs* est faux. Effectuez les étapes suivantes :
  1. [Signale l'exception](#) donnée par *evaluationStatus* .[[Value]] pour le *script* .
  2. [Nettoyer après avoir exécuté le script](#) avec *les paramètres* .
  3. Renvoyez *evaluationStatus* .
8. [Nettoyer après avoir exécuté le script](#) avec *les paramètres* .
9. Si *evaluationStatus* est un achèvement normal, retournez *evaluationStatus* .
10. Si nous avons atteint ce point, *evaluationStatus* a été laissé à null car le script a été [abandonné prématurément](#) lors de l'évaluation. Return Completion {  
[[Type]] : lancer, [[Valeur]] : un nouveau "[QuotaExceededError](#)" [DOMException](#) ,  
[[Cible]] : vide }.

Pour **exécuter un script de module** étant donné un *script* [de script de module](#) et un booléen facultatif *preventErrorReporting* (faux par défaut) :

1. Soit *settings* l' [objet settings](#) de *script* .
2. [Vérifiez si nous pouvons exécuter un script](#) avec *settings* . Si cela renvoie "ne pas exécuter", renvoyez [une promesse résolue avec](#) avec undefined.
3. [Préparez-vous à exécuter le script](#) avec *les paramètres* .
4. Soit *évaluationPromise* égal à null.
5. Si l'erreur du *script* [à relancer](#) n'est pas nulle, définissez *évaluationPromise* sur [une promesse rejetée avec](#) l'erreur du *script* [à relancer](#) .
6. Sinon:
  1. Soit *record* l' [enregistrement](#) du *script* .
  2. Définissez *évaluationPromise* sur *record* . [Évaluer](#) ().

*Cette étape évaluera de manière récursive toutes les dépendances du module.*

Si [Evaluate](#) ne se termine pas suite à [l'abandon du script en cours d'exécution](#) par l'agent utilisateur ,

définissez *evaluationPromise* sur [une promesse rejetée avec](#) un nouveau ["QuotaExceededError"](#) [DOMException](#) .

7. Si *preventErrorReporting* a la valeur false, lors [du rejet](#) de *evaluationPromise* avec *reason* , [signalez l'exception](#) donnée par *reason for script* .
8. [Nettoyer après avoir exécuté le script](#) avec *les paramètres* .
9. Retour *d'évaluationPromise* .

Les étapes pour **vérifier si nous pouvons exécuter un script** avec *les paramètres d'un [objet de paramètres d'environnement](#)* sont les suivantes. Ils renvoient soit "exécuter" soit "ne pas exécuter".

1. Si l' [objet global](#) spécifié par *les paramètres* est un [Window](#) objet dont [Document](#) l'objet n'est pas [complètement actif](#) , alors retournez "ne pas exécuter".
2. Si [le script est désactivé](#) pour *les paramètres* , alors renvoyez "ne pas exécuter".
3. Retourne "exécuter".

Les étapes pour **préparer l'exécution d'un script** avec *les paramètres d'un [objet de paramètres d'environnement](#)* sont les suivantes :

1. Poussez le contexte d' [exécution](#) du domaine des paramètres sur la [pile de contexte d'exécution JavaScript](#) ; c'est maintenant le [contexte d'exécution de JavaScript en cours d'exécution](#) .
2. Ajoutez *des paramètres* à [l'ensemble d'objets de paramètres d'environnement d'évaluation de script](#) de [la tâche](#) en cours d'exécution .

Les étapes de **nettoyage après l'exécution du script** avec *les paramètres d'un [objet de paramètres d'environnement](#)* sont les suivantes :

1. [Assert](#) : [le contexte d'exécution](#) du domaine *des paramètres* est le [contexte d'exécution JavaScript en cours d'exécution](#) .
2. Supprimez [le contexte d'exécution](#) du domaine des *paramètres* de la [pile de contexte d'exécution JavaScript](#) .
3. Si la [pile de contexte d'exécution JavaScript](#) est maintenant vide, [effectuez un point de contrôle de microtâche](#) . (Si cela exécute des scripts, ces algorithmes seront appelés de manière réentrante.)

*Ces algorithmes ne sont pas invoqués par un script en appelant directement un autre, mais ils peuvent être invoqués de manière réentrante de manière indirecte, par exemple si un script distribue un événement qui a des écouteurs d'événement enregistrés.*

Le **script en cours d'exécution** est le [script](#) du champ `[[HostDefined]]` du composant `ScriptOrModule` du [contexte d'exécution JavaScript en cours d'exécution](#) .

#### 8.1.4.5 Arrêter des scripts

Bien que la spécification JavaScript ne prenne pas en compte cette possibilité, il est parfois nécessaire d' **abandonner un script en cours d'exécution** . Cela entraîne l'arrêt immédiat de tout appel [ScriptEvaluation](#) ou [Source Text Module Record Evaluate](#) , vidant la pile de contexte d'exécution JavaScript sans déclencher aucun des mécanismes normaux tels que `finally` les blocs. [\[JAVASCRIPT\]](#)

Les agents utilisateurs peuvent imposer des limitations de ressources aux scripts, par exemple des quotas de processeur, des limites de mémoire, des limites de temps d'exécution total ou des limitations de bande passante. Lorsqu'un script dépasse une limite, l'agent utilisateur peut soit lancer un ["QuotaExceededError" DOMException](#) , [abandonner le script](#) sans exception, inviter l'utilisateur ou limiter l'exécution du script.

Par exemple, le script suivant ne se termine jamais. Un agent utilisateur pourrait, après avoir attendu quelques secondes, demander à l'utilisateur de terminer le script ou de le laisser continuer.

```
<script>
  while (true) { /* loop */ }
</script>
```

Les agents utilisateurs sont encouragés à autoriser les utilisateurs à désactiver les scripts chaque fois que l'utilisateur est invité soit par un script (par exemple en utilisant l' [window.alert\(\)](#) API) soit à cause des actions d'un script (par exemple parce qu'il a dépassé une limite de temps).

Si le script est désactivé pendant l'exécution d'un script, le script doit être arrêté immédiatement.

Les agents utilisateurs peuvent autoriser les utilisateurs à désactiver spécifiquement les scripts uniquement dans le but de fermer un [contexte de navigation](#) .

Par exemple, l'invite mentionnée dans l'exemple ci-dessus pourrait également offrir à l'utilisateur un mécanisme lui permettant de fermer entièrement la page, sans exécuter de [unload](#) gestionnaire d'événements.

#### 8.1.4.6 Erreurs de script d'exécution





`self.reportError(e)`

Distribue un `error` événement à l'objet global pour la valeur donnée `e`, de la même manière qu'une exception non gérée.

Lorsque l'agent utilisateur est tenu de **signaler une erreur** pour un script particulier *script* avec une *ligne* de position particulière : *col*, en utilisant une *cible* particulière *target*, il doit exécuter ces étapes, après quoi l'erreur est soit **gérée**, soit **non gérée** :

1. Si la *cible* est [en mode de rapport d'erreurs](#), alors retour ; l'erreur [n'est pas gérée](#).
2. Laissez la *cible* être **en mode de rapport d'erreurs**.
3. Soit *message* une chaîne [définie par l'implémentation](#) décrivant l'erreur de manière utile.
4. Soit *errorValue* la valeur qui représente l'erreur : dans le cas d'une exception non interceptée, ce serait la valeur qui a été levée ; dans le cas d'une erreur JavaScript, ce serait un `Error` objet. S'il n'y a pas de valeur correspondante, la valeur nulle doit être utilisée à la place.
5. Soit *urlString* le résultat de l'application du [séréaliseur d'URL](#) à l'[enregistrement d'URL](#) qui correspond à la ressource à partir de laquelle le *script* a été obtenu.

La ressource contenant le script sera généralement le fichier à partir duquel a `Document` été analysé, par exemple pour *script* les éléments en ligne ou [les attributs de contenu du gestionnaire d'événements](#) ; ou le fichier JavaScript dans lequel se trouvait le script, pour les scripts externes. Même pour les scripts générés dynamiquement, les agents utilisateurs sont fortement encouragés à essayer de garder une trace de la source originale d'un script. Par exemple, si un script externe utilise l'`document.write()` API pour insérer un *script* élément en ligne lors de l'analyse, l'URL de la ressource contenant le script serait idéalement signalée comme étant le script externe, et le numéro de ligne pourrait idéalement être signalé comme la ligne avec le `document.write()` call ou où la chaîne passée à cet appel a été construite en premier. Naturellement, la mise en œuvre de cela peut être quelque peu non triviale.

De même, les agents utilisateurs sont encouragés à suivre attentivement les numéros de ligne d'origine, même face à `document.write()` des appels modifiant le document lors de son analyse, ou [des attributs de contenu du gestionnaire d'événements](#) s'étendant sur plusieurs lignes.

6. Si *script* est un [script classique](#) et que [les erreurs masquées](#) du *script* sont vraies, définissez *message* sur "", *urlString* sur la chaîne vide, *line* et *col* sur 0 et *errorValue* sur `null.Script error`.



7. Soit *notHandled* vrai.
8. Si *la cible* implémente [EventTarget](#), définissez *notHandled* sur le résultat du [déclenchement d'un événement](#) nommé *error* sur *la cible*, en utilisant [ErrorEvent](#), avec l'attribut *cancelable* initialisé sur *true*, l'attribut *message* initialisé sur *message*, l'attribut *filename* initialisé sur *urlString*, l'attribut *lineno* initialisé sur *line*, l'attribut *colno* initialisé sur *col*, et l'attribut *error* initialisé à *errorValue*.
9. Ne laissez plus *la cible* être [en mode de rapport d'erreurs](#).
10. Si *notHandled* vaut *false*, alors l'erreur est [gérée](#). Sinon, l'erreur [n'est pas gérée](#).

*Retourner true dans un gestionnaire d'événements annule l'événement selon [l'algorithme de traitement du gestionnaire d'événements](#).*

Lorsque l'agent utilisateur doit **signaler une exception** *E*, l'agent utilisateur doit [signaler l'erreur pour le script](#) concerné, avec la position problématique (numéro de ligne et numéro de colonne) dans la ressource contenant le script, en utilisant l'[objet global spécifié par les paramètres](#) du script [objet](#) comme cible. Si l'erreur n'est toujours [pas traitée](#) après cela, l'erreur peut être signalée à une console de développeur.

L'existence à la fois [de signaler une erreur](#) et [de signaler une exception](#) prête à confusion, et les deux algorithmes ont des problèmes connus. Vous pouvez suivre le futur nettoyage de cette zone dans [le numéro 958](#).

Les étapes de la méthode consistent à [signaler l'exception](#) *e*.`reportError(e)`



L' [ErrorEvent](#) interface est définie comme suit :

```
[Exposed=*]
```

```
interface ErrorEvent : Event {
```

```
  constructor(DOMString type, optional ErrorEventInit
```

```
  eventInitDict = {});
```

```
  readonly attribute DOMString message;
```

```
  readonly attribute USVString filename;
```

```
readonly attribute unsigned long lineno;
```

```
readonly attribute unsigned long colno;
```

```
readonly attribute any error;
```

```
};
```

```
dictionary ErrorEventInit : EventInit {
```

```
    DOMString message = "";
```

```
    USVString filename = "";
```

```
    unsigned long lineno = 0;
```

```
    unsigned long colno = 0;
```

```
    any error;
```

```
};
```

L' **message** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente le message d'erreur.

L' **filename** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente l' [URL](#) du script dans lequel l'erreur s'est produite à l'origine.

L' **lineno** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente le numéro de ligne où l'erreur s'est produite dans le script.

L' **colno** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente le numéro de colonne où l'erreur s'est produite dans le script.

L' **error** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il doit initialement être initialisé à undefined. Le cas échéant, il est défini sur l'objet représentant l'erreur (par exemple, l'objet exception dans le cas d'une exception non interceptée).

#### 8.1.4.7 Refus de promesses non gérées

En plus des [erreurs de script d'exécution](#) synchrones, les scripts peuvent rencontrer des rejets de promesses asynchrones, suivis via les événements [unhandledrejection](#) et [rejectionhandled](#). Le suivi de ces rejets est effectué via l'opération abstraite [HostPromiseRejectionTracker](#), mais leur rapport est défini ici.

Pour **notifier les promesses rejetées** sur un *objet* [de paramètres d'environnement](#) donné :

1. Soit *list* une copie de la [liste des promesses rejetées sur le point d'être notifiées](#) de l'*objet paramètres*.
2. Si *la liste* est vide, retour.
3. Effacer la liste des promesses rejetées de l' *objet Paramètres* [sur le point d'être notifié](#).
4. Soit *global* l' [objet global](#) de l'*objet paramètres*.
5. [Mettez en file d'attente une tâche globale](#) sur la [source de tâche de manipulation DOM](#) donnée *globale* pour exécuter la sous-étape suivante :
  1. Pour chaque promesse *p* de *la liste* :
    1. Si le slot interne `[[PromiselsHandled]]` de *p* est vrai, passez à l'itération suivante de la boucle.
    2. Soit *notHandled* le résultat du [déclenchement d'un événement](#) nommé [unhandledrejection](#) global, en utilisant, avec l'attribut initialisé à `true`, l'attribut initialisé à *p* et l'attribut initialisé à la valeur de l'emplacement interne `[[PromiseResult]]` de *p*. [PromiseRejectionEventcancelablepromisereason](#)
    3. Si *notHandled* est false, alors le rejet de la promesse est [géré](#). Sinon, le rejet de la promesse n'est [pas géré](#).
    4. Si l'emplacement interne `[[PromiselsHandled]]` de *p* est *faux*, ajoutez *p* à [l'ensemble faible des promesses rejetées en attente](#) de l'*objet de paramètres*.

Cet algorithme a pour résultat que les refus de promesses sont marqués comme **traités** ou **non traités**. Ces concepts traitent en parallèle les erreurs de script [gérées](#) et [non gérées](#). Si un rejet n'est toujours [pas traité](#) après cela, le rejet peut être signalé à une console de développeur.



L' [PromiseRejectionEvent](#) interface est définie comme suit :

[Exposed=*]
-------------

```
interface PromiseRejectionEvent : Event {
```

```
  constructor(DOMString type, PromiseRejectionEventInit
```

```
  eventInitDict);
```

```
  readonly attribute Promise<any> promise;
```

```
  readonly attribute any reason;
```

```
};
```

```
dictionary PromiseRejectionEventInit : EventInit {
```

```
  required Promise<any> promise;
```

```
  any reason;
```

```
};
```



L' **promise** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente la promesse sur laquelle porte cette notification.



L' **reason** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente la raison du rejet de la promesse.

#### 8.1.4.8 Importer les résultats d'analyse de carte

Un **résultat d'analyse de carte d'importation** est une [structure](#) similaire à un [script](#) et peut également être stocké dans le [résultat](#)[script](#) d'un élément , mais n'est pas considéré comme un [script](#) à d'autres fins. Il comporte les [éléments](#) suivants :

##### Une *carte d'importation*

Une [carte d'importation](#) ou null.

##### Une *erreur à relancer*

Une valeur JavaScript représentant une erreur qui empêchera l'utilisation de cette carte d'importation, lorsqu'elle n'est pas nulle.

Pour **créer un résultat d'analyse de carte d'importation** avec une *entrée* [de chaîne](#) et une [URL](#) *baseURL* :

1. Soit *result* un [résultat d'analyse de carte d'importation](#) dont [la carte d'importation](#) est nulle et dont [l'erreur à relancer](#) est nulle.
2. [Analysez une chaîne de mappage d'importation](#) en fonction de *l'entrée* et de *la baseURL* , en interceptant toutes les exceptions. Si cela a levé une exception, alors définissez l'erreur du *résultat* [pour relancer](#) cette exception. Sinon, définissez [la carte d'importation](#) du *résultat* sur la valeur de retour.
3. Retourner *le résultat* .

Pour **enregistrer une mappe d'importation** à partir d'un *résultat d'analyse* [window](#) *globale* et d'une [mappe d'importation](#) :

1. Si l'erreur de *résultat* [à relancer](#) n'est pas nulle, [signalez l'exception](#) donnée par l'erreur de *résultat* [à relancer](#) et à renvoyer.
2. [Assert](#) : [la carte d'importation](#) de *global* est une [carte d'importation vide](#) .
3. Définissez [la carte d'importation](#) globale sur la [carte d'importation](#) du *résultat* .

## 8.1.5 Résolution du spécificateur de module

### 8.1.5.1 L'algorithme de résolution

L' algorithme [de résolution d'un spécificateur de module](#) est le principal point d'entrée pour convertir les chaînes de spécificateur de module en [URL](#) . Lorsqu'aucune [carte d'importation](#) n'est impliquée, cela est relativement simple et se résume à [la résolution d'un spécificateur de module de type URL](#) .

Lorsqu'une [carte d'importation](#) non vide est présente, le comportement est plus complexe. Il vérifie les entrées candidates à partir de toutes [les mappes de spécificateurs de module applicables, des portées](#) les plus spécifiques aux moins spécifiques (revenant aux [importations](#) sans portée de niveau supérieur ) et des préfixes les plus spécifiques aux moins spécifiques. Pour chaque candidat, la [résolution d'un](#) algorithme de correspondance des importations donnera les résultats suivants :

- Résolution réussie du spécificateur en [URL](#) . Ensuite, l' algorithme [de résolution d'un spécificateur de module](#) renverra cette URL.

- Lancer une exception. Ensuite, l'algorithme [de résolution d'un spécificateur de module](#) lèvera à nouveau cette exception, sans autre recours.
- Échec de la résolution, sans erreur. Dans ce cas, l'algorithme [de résolution externe d'un spécificateur de module](#) passera au candidat suivant.

En fin de compte, si aucune résolution réussie n'est trouvée via l'un des [mappages de spécificateurs de module](#) candidats, [résoudre un spécificateur de module](#) lèvera une exception. Ainsi, le résultat est toujours soit une [URL](#), soit une exception levée.

Pour **résoudre un spécificateur de module** donné un [script](#) -or-null *referringScript* et un *spécificateur* [de chaîne](#) :

1. Laissez *settingsObject* et *baseURL* être nuls.
2. Si *referringScript* n'est pas nul, alors :
  1. Définissez *settingsObject* sur [l'objet settings](#) de *referringScript*.
  2. Définissez *baseURL* sur [l'URL](#) de base de *referringScript*.
3. Sinon:
  1. [Assert](#) : il existe un [objet de paramètres actuel](#).
  2. Définissez *settingsObject* sur l' [objet de paramètres actuel](#).
  3. Définissez *baseURL* sur [l'URL de base de l'API](#) de *settingsObject*.
4. Soit *importMap* une [carte d'importation vide](#).
5. Si [l'objet global](#) de *settingsObject* implémente, définissez *importMap* sur [la carte d'importation](#) de l' objet [global](#) de *settingsObject* [.Window](#).
6. Soit *baseURLString* être *baseURL*, [sérialisé](#).
7. Soit *asURL* le résultat de [la résolution d'un spécificateur de module de type URL](#) donné *specifier* et *baseURL*.
8. Soit *normalizedSpecifier* la [sérialisation](#) de *asURL*, si *asURL* n'est pas null ; sinon, *spécificateur*.
9. [Pour chaque](#) *scopePrefix* → *scopeImports* des [scopes](#) de *importMap* :
  1. Si *scopePrefix* est *baseURLString*, ou si *scopePrefix* se termine par U+002F (/) et *scopePrefix* est un [préfixe d'unité de code](#) de *baseURLString*, alors :
    1. Soit *scopeImportsMatch* le résultat de [la résolution d'une correspondance d'importations](#) en fonction de *normalizedSpecifier*, *asURL* et *scopeImports*.

2. Si `scopeImportsMatch` n'est pas null, retournez `scopeImportsMatch`.
10. Supposons que `topLevelImportsMatch` soit le résultat de la résolution d' [une correspondance d'importation](#) en fonction des [importations](#) de `normalizedSpecifieur`, `asURL` et `importMap`.
11. Si `topLevelImportsMatch` n'est pas nul, alors renvoie `topLevelImportsMatch`.
12. À ce stade, le spécificateur n'a été remappé sur rien par `importMap`, mais il aurait peut-être pu être transformé en URL.

Si `asURL` n'est pas nul, alors renvoie `asURL`.

13. Lancez un `TypeError` indiquant que le spécificateur était un spécificateur nu, mais qu'il n'a été remappé sur rien par `importMap`.

Pour **résoudre une correspondance d'importation**, étant donné une [chaîne](#) `normalizedSpecifieur`, une [URL](#) -ou-null `asURL` et une [mappe de spécificateur de module](#) `specifieurMap`:

1. [Pour chaque](#) `specifieurKey` → `resolutionResult` de `specifieurMap`:
    1. Si `specifieurKey` est `normalizedSpecifieur`, alors :
      1. Si `resolutionResult` est null, lancez un `TypeError` indiquant que la résolution de `specifieurKey` a été bloquée par une entrée null.  

[Cela mettra fin à l'intégralité de la résolution d'un algorithme de spécification de module](#), sans autre recours.
      2. [Assert](#) : `resolutionResult` est une [URL](#).
      3. Retourne `resolutionResult`.
    2. Si toutes les conditions suivantes sont vraies :
      1. `specifieurKey` se termine par U+002F (/);
      2. `specifieurKey` est un [préfixe d'unité de code](#) de `normalizedSpecifieur`; et
      3. soit `asURL` est null, soit `asURL` [est special](#),
- alors:
4. Si `resolutionResult` est null, lancez un `TypeError` indiquant que la résolution de `specifieurKey` a été bloquée par une entrée null.  

[Cela mettra fin à l'intégralité de la résolution d'un algorithme de spécification de module](#), sans autre recours.
  5. [Assert](#) : `resolutionResult` est une [URL](#).

6. Soit *afterPrefix* la partie de *normalizedSpecifieur* après le préfixe initial *specifieurKey* .
7. Assert : *resolutionResult* , sérialisé , se termine par U+002F (/), tel qu'appliqué lors de l'analyse .
8. Soit *url* le résultat de l'analyse d'URL *afterPrefix* avec *resolutionResult* .
9. Si *url* est un échec, lancez un `TypeError` indiquant que la résolution de *normalizedSpecifieur* a été bloquée car la partie *afterPrefix* n'a pas pu être analysée par URL par rapport au *resolutionResult* mappé par le préfixe *specifieurKey* .

*Cela mettra fin à l'intégralité de la résolution d'un algorithme de spécification de module , sans autre recours.*

10. Assert : *url* est une URL .
11. Si la sérialisation de *resolutionResult* n'est pas un préfixe d'unité de code de la sérialisation de *url* , lancez un `TypeError` indiquant que la résolution de *normalizedSpecifieur* a été bloquée en raison de son retour en arrière au-dessus de son préfixe *specifieurKey* .

*Cela mettra fin à l'intégralité de la résolution d'un algorithme de spécification de module , sans autre recours.*

12. *URL* de retour .

## 2. Renvoi nul.

*L' algorithme de résolution d'un spécificateur de module reviendra à une portée moins spécifique, ou à " imports", si possible.*

Pour **résoudre un spécificateur de module de type URL** , étant donné un *spécificateur de chaîne* et une URL *baseURL* :

1. Si le *spécificateur* commence par " /", " ./" ou " ../", alors :
  1. Soit *url* le résultat du *spécificateur d'analyse d'URL* avec *baseURL* .
  2. Si *url* est un échec, alors retournez null.

Cela pourrait se produire si le *spécificateur* est " ../foo" et *baseURL* est une `data:`URL.

3. *URL* de retour .

*Cela inclut les cas où le spécificateur commence par " /", c'est-à-dire les URL relatives au schéma. Ainsi, url peut se retrouver avec un hôte différent de *baseURL* .*



2. Soit *url* le résultat du *spécificateur d'analyse d'URL* (sans URL de base).
3. Si *url* est un échec, alors retournez null.
4. *URL* de retour .

### 8.1.5.2 Importer des cartes

Une [mappe d'importation](#) permet de contrôler la résolution du spécificateur de module. Les cartes d'importation sont livrées via [script](#) des éléments en ligne avec leur [type](#) attribut défini sur "importmap", et avec leur [contenu textuel enfant](#) contenant une représentation JSON de la carte d'importation.

Une seule carte d'importation est traitée par fichier [Document](#). Une fois que la première carte d'importation est vue, les autres seront ignorées, leurs [script](#) éléments correspondants générant [error](#) des événements. De même, une fois que tous les modules ont été importés, par exemple, via [import\(\)](#) des expressions ou [script](#) des éléments avec leur [type](#) attribut défini sur "module", d'autres cartes d'importation seront ignorées.

*Ces restrictions, ainsi que le manque de prise en charge des cartes d'importation externes, sont en place pour garder la version initiale de la fonctionnalité simple. Ils peuvent être levés au fil du temps si la bande passante de l'implémenteur le permet.* L'utilisation la plus simple des cartes d'importation consiste à remapper globalement un spécificateur de module nu :

```
{
  "imports": {
    "moment": "/node_modules/moment/src/moment.js"
  }
}
```

Cela permet à des déclarations telles que `import moment from "moment";` de fonctionner, de récupérer et d'évaluer le module JavaScript à l'URL `/node_modules/moment/src/moment.js`.

Un mappage d'importation peut remapper une classe de spécificateurs de module dans une classe d'URL en utilisant des barres obliques à la fin, comme ceci :

```
{
  "imports": {
    "moment/": "/node_modules/moment/src/"
  }
}
```

Cela permet à des déclarations telles que `import localeData from "moment/locale/zh-cn.js";` de fonctionner, de récupérer et d'évaluer le module JavaScript à l'URL `/node_modules/moment/src/moment/locale/zh-cn.js`. Ces mappages de barre oblique de fin sont souvent combinés avec des mappages de spécificateur nu, par exemple

```
{
  "imports": {
    "moment": "/node_modules/moment/src/moment.js",
    "moment/": "/node_modules/moment/src/"
  }
}
```

de sorte que le "module principal" spécifié par `"moment"` et les "sous-modules" spécifiés par des chemins tels que `"moment/locale/zh-cn.js"` soient disponibles.

Les spécificateurs nus ne sont pas le seul type de spécificateurs de module que les cartes d'importation peuvent remapper. Les spécificateurs "de type URL", c'est-à-dire ceux qui sont analysables comme des URL absolues ou qui commencent par `" /`, `" ./` ou `" ../`, peuvent également être remappés :

```
{
  "imports": {
    "https://cdn.example.com/vue/dist/vue.runtime.esm.js":
      "/node_modules/vue/dist/vue.runtime.esm.js",
    "/js/app.mjs": "/js/app-8e0d62a03.mjs",
    "../helpers/": "https://cdn.example/helpers/"
  }
}
```

Notez que l'URL à remapper, ainsi que l'URL à mapper, peuvent être spécifiées soit en tant qu'URL absolues, soit en tant qu'URL relatives commençant par `" /`, `" ./` ou `" ../`. (Ils ne peuvent pas être spécifiés en tant qu'URL relatives sans ces sigils de départ, car ceux-ci permettent de les distinguer des spécificateurs de module nus.) Notez également comment le [mappage de la barre oblique finale](#) fonctionne également dans ce contexte.

Ces remappages fonctionnent sur l'URL post-canonisation et ne nécessitent pas de correspondance entre les chaînes littérales fournies dans la clé de mappage d'importation et le spécificateur de module importé. Ainsi, par exemple, si cette carte d'importation était incluse dans `https://example.com/app.html`, non seulement elle serait `import "/js/app.mjs"` remappée, mais aussi `import "../js/app.mjs"` et `import "../foo/../js/app.mjs"`.

Tous les exemples précédents ont des spécificateurs de module remappés globalement, en utilisant la clé `" "` de niveau supérieur `imports` dans le mappage d'importation. La clé `" "` de niveau supérieur `scopes` peut être utilisée pour fournir des

remappings localisés, qui ne s'appliquent que lorsque le module de référence correspond à un préfixe d'URL spécifique. Par exemple:

```
{
  "scopes": {
    "/a/" : {
      "moment": "/node_modules/moment/src/moment.js"
    },
    "/b/" : {
      "moment": "https://cdn.example.com/moment/src/moment.js"
    }
  }
}
```

Avec cette mappe d'importation, l'instruction `import "moment"` aura des significations différentes selon le script de référence qui contient l'instruction :

- Dans les scripts situés sous `/a/`, cela importera `/node_modules/moment/src/moment.js`.
- Dans les scripts situés sous `/b/`, cela importera `https://cdn.example.com/moment/src/moment.js`.
- Dans les scripts situés sous `/c/`, cela échouera à résoudre et lèvera donc une exception.

Une utilisation typique des portées est de permettre à plusieurs versions du "même" module d'exister dans une application Web, certaines parties du graphique du module important une version et d'autres parties important une autre version.

Les portées peuvent se chevaucher et chevaucher la `imports` carte de spécificateur " globale. Au moment de la résolution, les étendues sont consultées dans l'ordre du plus spécifique au moins spécifique, la spécificité étant mesurée en triant les étendues à l'aide de l' [unité de code inférieure à](#) l'opération. Ainsi, par exemple, `" /scope2/scope3/"` est traité comme plus spécifique que `" /scope2/"`, qui est traité comme plus spécifique que les mappages de niveau supérieur (non délimités).

La carte d'importation suivante illustre cela :

```
{
  "imports": {
    "a": "/a-1.mjs",
    "b": "/b-1.mjs",
    "c": "/c-1.mjs"
  },
  "scopes": {
```

```

    "/scope2/": {
      "a": "/a-2.mjs"
    },
    "/scope2/scope3/": {
      "b": "/b-3.mjs"
    }
  }
}

```

Cela se traduit par les résolutions suivantes (en utilisant des URL relatives par souci de brièveté) :

		Prescripteur		
		" a "	" b "	" c "
Réfèrent	/scope1/r.mjs	/a-1.mjs	/b-1.mjs	/c-1.mjs
	/scope2/r.mjs	/a-2.mjs	/b-1.mjs	/c-1.mjs
	/scope2/scope3/r.mjs	/a-2.mjs	/b-3.mjs	/c-1.mjs

Le [contenu du texte enfant](#) d'un [script](#) élément représentant une [mappe d'importation doit correspondre aux exigences de création de mappe d'importation](#) suivantes :

- Il doit s'agir d'un JSON valide. [\[JSON\]](#)
- Le JSON doit représenter un objet JSON, avec au plus les deux clés " imports " et " scopes ".
- Les valeurs correspondant aux clés " imports " et " scopes ", si présentes, doivent elles-mêmes être des objets JSON.
- La valeur correspondant à la imports clé " ", si elle est présente, doit être une [mappe de spécificateur de module valide](#) .
- La valeur correspondant à la scopes clé " ", si elle est présente, doit être un objet JSON, dont les clés sont [des chaînes d'URL valides](#) et dont les valeurs sont [des cartes de spécificateur de module valides](#) .

Un **mappage de spécificateur de module valide** est un objet JSON qui répond aux exigences suivantes :

- Toutes ses clés doivent être non vides.
- Toutes ses valeurs doivent être des chaînes.

- Chaque valeur doit être soit une [URL absolue valide](#) , soit une [chaîne d'URL valide](#) commençant [par](#) " /", " ./" ou " ../".
- Si une clé donnée [se termine par](#) " /", alors la valeur correspondante doit également.

### 8.1.5.3 Importer le modèle de traitement de carte

Formellement, une **carte d'importation** est une [structure](#) à deux [éléments](#) :

- **importe** , une [carte de spécificateur de module](#) ; et
- **scopes** , une [carte ordonnée](#) d' [URL](#) vers [des cartes de spécificateurs de module](#) .

Une **mappe de spécificateur de module** est une [mappe ordonnée](#) dont [les clés](#) sont [des chaînes](#) et dont [les valeurs](#) sont soit [des URL](#) , soit des valeurs nulles.

Une **mappe d'importation vide** est une [mappe d'importation](#) dont [les importations](#) et [les étendues](#) sont toutes deux des mappes vides.

---

Chacun [Window](#) a une **carte d'importation** , initialement une [carte d'importation vide](#) .

Chacun [Window](#) a une **carte d'importation autorisée** booléen, initialement vrai.

Pour **interdire d'autres cartes d'importation** étant donné un [objet de paramètres d'environnement](#) `settingsObject` :

1. Soit `global` l' [objet global](#) de `settingsObject` .
2. Si `global` n'implémente pas [Window](#), alors retournez.
3. Définissez les cartes d'importation de `global` [autorisées](#) sur false.

*Les cartes d'importation sont actuellement interdites une fois qu'un chargement de module a commencé ou une fois qu'une seule carte d'importation est chargée. Ces restrictions pourraient être levées dans les futures révisions des spécifications.*

---

Pour **analyser une chaîne de carte d'importation** , étant donné une *entrée de chaîne* et une *URL baseURL* :

1. Soit *parsed* le résultat de [l'analyse d'une chaîne JSON en une valeur Infra](#) donnée *input* .
  2. Si *parsed* n'est pas une [carte ordonnée](#) , lancez un `TypeError` indiquant que la valeur de niveau supérieur doit être un objet JSON.
  3. Soit *sortedAndNormalizedImports* une [carte ordonnée](#) vide .
  4. Si `[ " " ] analysé existe` , alors `:imports`
    1. Si `[ " " ] analysé imports` n'est pas une [carte ordonnée](#) , lancez un `TypeError` indiquant que la valeur de la `imports` clé de niveau supérieur `" "` doit être un objet JSON.
    2. Définissez *sortedAndNormalizedImports* sur le résultat du [tri et de la normalisation d'un mappage de spécificateur de module](#) donné *parsed* `[ " imports" ]` et *baseURL* .
  5. Soit *sortedAndNormalizedScopes* une [carte ordonnée](#) vide .
  6. Si `[ " " ] analysé existe` , alors `:scopes`
    1. Si `[ " " ] analysé scopes` n'est pas une [carte ordonnée](#) , lancez un `TypeError` indiquant que la valeur de la `scopes` clé de niveau supérieur `" "` doit être un objet JSON.
    2. Définissez *sortedAndNormalizedScopes* sur le résultat du [tri et de la normalisation des portées](#) analysées `[ " " scopes ]` et *baseURL* .
  7. Si [les clés](#) de *parsed* [contiennent](#) des éléments autres que `" "` ou `" "` , l'agent utilisateur doit [signaler un avertissement à la console](#) indiquant qu'une clé de niveau supérieur invalide était présente dans la carte d'importation. `importssscopes`
- Cela peut aider à détecter les fautes de frappe. Ce n'est pas une erreur, car cela empêcherait toute future extension d'être ajoutée de manière rétrocompatible.*
8. Renvoie un [mappage d'importation](#) dont [les importations](#) sont *triéesAndNormalizedImports* et dont [les portées](#) sont *triéesAndNormalizedScopes* .

La [carte d'importation](#) qui résulte de cet algorithme d'analyse est hautement normalisée. Par exemple, étant donné une URL de base de `https://example.com/base/page.html`, l'entrée

```
{  
  "imports": {
```

```

    "/app/helper": "node_modules/helper/index.mjs",
    "lodash": "/node_modules/lodash-es/lodash.js"
  }
}

```

générera une [carte d'importation](#) avec des [importations](#) de

```

« [
  "https://example.com/app/helper" →
  https://example.com/base/node_modules/helper/index.mjs
  "lodash" → https://example.com/node_modules/lodash-es/lodash.js
] »

```

et (bien que rien ne soit présent dans la chaîne d'entrée) une [carte ordonnée](#) vide pour ses [scopes](#) .

Pour **trier et normaliser une carte de spécificateur de module** , étant donné une [carte ordonnée](#) *originalMap* et une [URL](#) *baseURL* :

1. Soit *normalized* une [carte ordonnée](#) vide .
2. [Pour chaque](#) *specifierKey* → *valeur* de *originalMap* :
  1. Soit *normalizedSpecifierKey* le résultat de [la normalisation d'une clé de spécification](#) donnée *specifierKey* et *baseURL* .
  2. Si *normalizedSpecifierKey* est null, [continuez](#) .
  3. Si *value* n'est pas une [chaîne](#) , alors :
    1. L'agent utilisateur peut [signaler un avertissement à la console](#) indiquant que les adresses doivent être des chaînes.
    2. Définissez *normalized* [ *normalizedSpecifierKey* ] sur null.
    3. [Continuez](#) .
  4. Soit *addressURL* le résultat de [la résolution d'un spécificateur de module de type URL](#) donné *value* et *baseURL* .
  5. Si *addressURL* est nul, alors :
    1. L'agent utilisateur peut [signaler un avertissement à la console](#) indiquant que l'adresse n'était pas valide.
    2. Définissez *normalized* [ *normalizedSpecifierKey* ] sur null.
    3. [Continuez](#) .

6. Si *specifierKey* se termine par U+002F (/) et que la sérialisation de *addressURL* ne se termine pas par U+002F (/), alors :
  1. L'agent utilisateur peut signaler un avertissement à la console indiquant qu'une adresse non valide a été donnée pour la clé de spécification *specifierKey* ; puisque *specifierKey* se termine par une barre oblique, l'adresse doit également l'être.
  2. Définissez *normalized* [ *normalizedSpecifierKey* ] sur null.
  3. Continuez .
7. Définissez *normalized* [ *normalizedSpecifierKey* ] sur *addressURL* .
3. Renvoie le résultat du tri dans l'ordre décroissant *normalisé* , une entrée *a* étant inférieure à une entrée *b* si la clé de *a* est une unité de code inférieure à la clé de *b* .

Pour **trier et normaliser les étendues** , étant donné une carte ordonnée *originalMap* et une URL *baseURL* :

1. Soit *normalized* une carte ordonnée vide .
2. Pour chaque *scopePrefix* → *potentialSpecifierMap* de *originalMap* :
  1. Si *potentialSpecifierMap* n'est pas une carte ordonnée , lancez un TypeError indiquant que la valeur de la portée avec le préfixe *scopePrefix* doit être un objet JSON.
  2. Soit *scopePrefixURL* le résultat de l'analyse d'URL *scopePrefix* avec *baseURL* .
  3. Si *scopePrefixURL* échoue, alors :
    1. L'agent utilisateur peut signaler un avertissement à la console indiquant que l'URL du préfixe de portée n'était pas analysable.
    2. Continuez .
  4. Soit *normalizedScopePrefix* la sérialisation de *scopePrefixURL* .
  5. Définissez *normalized* [ *normalizedScopePrefix* ] sur le résultat du tri et de la normalisation d'une carte de spécificateur de module en fonction de *potentialSpecifierMap* et *baseURL* .
3. Renvoie le résultat du tri dans l'ordre décroissant *normalisé* , une entrée *a* étant inférieure à une entrée *b* si la clé de *a* est une unité de code inférieure à la clé de *b* .



*Dans les deux algorithmes ci-dessus, le tri des clés et des portées par ordre décroissant a pour effet de placer "foo/bar/" avant "foo/". Cela donne à son tour "foo/bar/" une priorité plus élevée que "foo/" lors de la résolution du spécificateur de module .*

Pour **normaliser une clé de spécification** , étant donné une chaîne *specifierKey* et une URL *baseURL* :

1. Si *specifierKey* est la chaîne vide, alors :
  1. L'agent utilisateur peut signaler un avertissement à la console indiquant que les clés de spécification peuvent ne pas être la chaîne vide.
  2. Renvoie nul.
2. Soit *url* le résultat de la résolution d'un spécificateur de module de type URL , étant donné *specifierKey* et *baseURL* .
3. Si *url* n'est pas null, renvoie la sérialisation de *url* .
4. Renvoie *specifierKey* .

### 8.1.6 Crochets d'hôte de spécification JavaScript

La spécification JavaScript contient un certain nombre d'opérations abstraites définies par l'implémentation , qui varient en fonction de l'environnement hôte. Cette section les définit pour les hôtes d'agent utilisateur.

#### 8.1.6.1 *HostEnsureCanAddPrivateElement* ( *O* )

JavaScript contient une opération abstraite *HostEnsureCanAddPrivateElement* ( *O* ) définie par l'implémentation . Les agents utilisateurs doivent utiliser l'implémentation suivante : [JAVASCRIPT]

1. Si *O* est un WindowProxy objet, ou implements Location , alors retourne Completion { [[Type]]: throw, [[Value]]: a new TypeError }.
2. Renvoie NormalCompletion (inutilisé).

*Les champs privés JavaScript peuvent être appliqués à des objets arbitraires. Étant donné que cela peut compliquer considérablement la mise en œuvre d'objets hôtes particulièrement exotiques, la spécification du langage JavaScript fournit ce crochet pour permettre aux hôtes de rejeter les champs privés sur les objets répondant à des critères définis par l'hôte. Dans le cas de HTML, WindowProxy et Location ont une sémantique compliquée - en particulier autour de la navigation et de la sécurité - qui*

*rend difficile l'implémentation de la sémantique des champs privés, donc notre implémentation rejette simplement ces objets.*

#### **8.1.6.2 *HostEnsureCanCompileStrings* ( domaine )**

JavaScript contient une opération abstraite [HostEnsureCanCompileStrings](#) ( *realm* ) [définie par l'implémentation](#) . Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

1. Effectuer ? [EnsureCSPDoesNotBlockStringCompilation](#) ( *domaine* ). [\[CSP\]](#)

#### **8.1.6.3 *HostPromiseRejectionTracker* ( promesse , opération )**

JavaScript contient une opération abstraite [HostPromiseRejectionTracker](#) ( *promise* , *operation* ) [définie par l'implémentation](#) . Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

1. Soit *script* le [script en cours d'exécution](#) .
2. Si le *script* est un [script classique](#) et que [les erreurs masquées](#) du *script* sont vraies, alors retournez.
3. Soit l'*objet de paramètres* l' [objet de paramètres actuel](#) .
4. Si *script* n'est pas nul, définissez l'*objet de paramètres* sur [l'objet de paramètres](#) du *script* .
5. Si l'*opération* est " *reject*", alors :
  1. [Ajouter](#) la promesse à [la liste des promesses rejetées sur le point d'être notifiées](#) de l'*objet de paramètres* .
6. Si l'*opération* est " *handle*", alors :
  1. Si [la liste des promesses rejetées sur le point d'être notifiées](#) de l'*objet de paramètres* [contient](#) promesse , [supprimez](#) la promesse de cette liste et revenez.
  2. Si [l'ensemble faible des promesses rejetées en suspens](#) de l'*objet de paramètres* ne [contient pas](#) de promesse , alors retournez.
  3. [Supprimer](#) la promesse de l'ensemble [faible](#) des promesses rejetées en attente de l'*objet de paramètres* .

4. Soit *global* l' [objet global](#) de l'objet *paramètres* .
5. [Mettez en file d'attente une tâche globale](#) sur la [source de la tâche de manipulation DOM](#) donnée *global* pour [déclencher un événement](#) nommé `rejectionhandledglobal` , en utilisant , avec l' attribut initialisé à *promise* , et l' *attribut* initialisé à *promise* .[[PromiseResult]].`PromiseRejectionEventpromiseReason`

#### 8.1.6.4 Crochets d'hôte liés à la tâche



La spécification JavaScript définit les travaux à planifier et à exécuter ultérieurement par l'hôte, ainsi que [les enregistrements JobCallback](#) qui encapsulent les fonctions JavaScript appelées dans le cadre des travaux. La spécification JavaScript contient un certain nombre d'opérations abstraites [définies par l'implémentation](#) qui permettent à l'hôte de définir comment les travaux sont planifiés et comment les JobCallbacks sont gérés. HTML utilise ces opérations abstraites pour suivre l' [objet de paramètres en place](#) dans les promesses et `FinalizationRegistry` les rappels en enregistrant et en restaurant l' [objet de paramètres en place](#) et un [contexte d'exécution JavaScript](#) pour le [script actif](#).dans JobCallbacks. Cette section les définit pour les hôtes d'agent utilisateur.

##### 8.1.6.4.1 HostCallJobCallback ( *callback* , *V* , *argumentsList* )

JavaScript contient une opération abstraite [HostCallJobCallback](#) ( *callback* , *V* , *argumentsList* ) [définie par l'implémentation pour permettre aux hôtes de restaurer l'état lors de l'appel de rappels JavaScript à partir de tâches internes](#). Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

1. Laisser *les paramètres en place* être *rappelés* .[[HostDefined]].[[IncumbentSettings]].
2. Laissez le *contexte d'exécution du script* être *callback* .[[HostDefined]].[[ActiveScriptContext]].
3. [Préparez-vous à exécuter un rappel](#) avec *les paramètres du titulaire* .

*Cela affecte le concept [de titulaire](#) pendant que le rappel s'exécute.*

4. Si le *contexte d'exécution du script* n'est pas nul, [placez](#) le *contexte d'exécution du script* dans la [pile de contexte d'exécution JavaScript](#) .

*Cela affecte le [script actif](#) pendant l'exécution du rappel.*

5. Soit *result* être [Call](#) ( *callback* .[[Callback]], *V* , *argumentsList* ).
6. Si le contexte d'exécution du script n'est pas nul, alors [pop](#) le contexte d'exécution du script de la [pile de contexte d'exécution JavaScript](#) .
7. [Nettoyez après avoir exécuté un rappel](#) avec les paramètres en place .
8. Retourner le résultat .

#### 8.1.6.4.2 *HostEnqueueFinalizationRegistryCleanupJob* ( *finalizationRegistry* )

JavaScript a la capacité d'enregistrer des objets avec [FinalizationRegistry](#) des objets, afin de programmer une action de nettoyage s'il s'avère qu'ils sont ramassés. La spécification JavaScript contient une opération abstraite [HostEnqueueFinalizationRegistryCleanupJob](#) ( *finalizationRegistry* ) définie par l'implémentation pour planifier l'action de nettoyage .

*Le moment et l'occurrence du travail de nettoyage sont [définis par l'implémentation](#) dans la spécification JavaScript. Les agents utilisateurs peuvent différer quant au moment et au fait qu'un objet soit ramassé ou non, affectant à la fois si la valeur de retour de la [WeakRef.prototype.deref\(\)](#) méthode est indéfinie et si [FinalizationRegistry](#) des rappels de nettoyage se produisent. Il existe des cas bien connus dans les navigateurs Web populaires où les objets ne sont pas accessibles à JavaScript, mais ils restent indéfiniment conservés par le ramasse-miettes. HTML efface les objets maintenus actifs [WeakRef](#) lors de [l'exécution d'un algorithme de point de contrôle de microtâche](#) . Les auteurs feraient mieux de ne pas dépendre des détails de synchronisation des implémentations de la récupération de place.*

Les actions de nettoyage ne sont pas intercalées avec l'exécution JavaScript synchrone, mais se produisent plutôt dans [les tâches](#) en file d'attente . Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

1. Soit *global* l' [objet global](#) *finalizationRegistry* .[[Realm]] .
2. [Mettez en file d'attente une tâche globale](#) sur la **source de tâche du moteur JavaScript** donnée *globale* pour effectuer les étapes suivantes :
  1. Soit l'entrée *finalizationRegistry* .[[CleanupCallback]].[[Callback]].[[Realm]]'s *environment* [settings object](#) .
  2. [Vérifiez si nous pouvons exécuter le script](#) avec l'entrée . Si cela renvoie "ne pas exécuter", alors revenez.
  3. [Préparez-vous à exécuter le script](#) avec l'entrée .

*Cela affecte le concept [d'entrée](#) pendant l'exécution du rappel de nettoyage.*

4. Soit *result* le résultat de l'exécution de [CleanupFinalizationRegistry](#) ( *finalizationRegistry* ).
5. [Nettoyer après avoir exécuté le script](#) avec l'entrée .
6. Si *result* est un [achèvement brutal](#) , alors [signalez l'exception](#) donnée par *result* .[[Value]].

#### 8.1.6.4.3 HostEnqueuePromiseJob ( tâche , domaine )

JavaScript contient une opération abstraite [HostEnqueuePromiseJob](#) ( *job* , *realm* ) [définie par l'implémentation](#) pour planifier les opérations liées à Promise. HTML planifie ces opérations dans la file d'attente des microtâches. Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

1. Si *realm* n'est pas null, laissez les paramètres du travail être l' [objet de paramètres](#) pour *realm* . Sinon, laissez les paramètres du travail être nuls.

*Si le domaine n'est pas nul, c'est le [domaine](#) du code de l'auteur qui s'exécutera. Lorsque *job* est renvoyé par [NewPromiseReactionJob](#) , il s'agit du domaine de la fonction de gestionnaire de la promesse. Lorsque *job* est renvoyé par [NewPromiseResolveThenableJob](#) , il s'agit du domaine de la `then` fonction.*

*Si le domaine est nul, soit aucun code d'auteur ne s'exécutera, soit le code d'auteur est garanti à lancer. Pour le premier, l'auteur peut ne pas avoir transmis de code à exécuter, comme dans `promise.then(null, null)`. Pour ces derniers, c'est parce qu'une procuration révoquée a été passée. Dans les deux cas, toutes les étapes ci-dessous qui utiliseraient autrement les paramètres de la tâche sont ignorées.*

2. [Mettez une microtâche en file d'attente](#) sur la [boucle d'événements](#) de l'[agent environnant](#) pour effectuer les étapes suivantes :

1. Si les paramètres du travail ne sont pas nuls, [vérifiez si nous pouvons exécuter le script](#) avec les paramètres du travail . Si cela renvoie "ne pas exécuter", alors revenez.
2. Si les paramètres du travail ne sont pas nuls, [préparez-vous à exécuter le script](#) avec les paramètres du travail .

*Cela affecte le concept [d'entrée](#) pendant l'exécution du travail.*

3. Soit *résultat* *job* ( ) .

**job* est une [fermeture abstraite](#) renvoyée par [NewPromiseReactionJob](#) ou [NewPromiseResolveThenableJob](#) . La fonction de gestionnaire de la promesse lorsque *job* est renvoyé*

par [NewPromiseReactionJob](#) , et la `then` fonction lorsque job est renvoyé par [NewPromiseResolveThenableJob](#) , sont encapsulées dans [JobCallback Records](#) . HTML enregistre l' [objet de paramètres en place](#) et un [contexte d'exécution JavaScript](#) pour le [script actif](#) dans [HostMakeJobCallback](#) et les restaure dans [HostCallJobCallback](#) .

4. Si les paramètres du travail ne sont pas nuls, [nettoyez après avoir exécuté le script](#) avec les paramètres du travail .
5. Si `result` est un [achèvement brutal](#) , alors [signalez l'exception](#) donnée par `result.[[Value]]`.

#### 8.1.6.4.4 HostMakeJobCallback ( *appelable* )

JavaScript contient une opération abstraite [HostMakeJobCallback](#) ( *callable* ) [définie par l'implémentation pour permettre aux hôtes d'attacher l'état aux rappels JavaScript qui sont appelés à partir des tâches](#) internes . Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

1. Laissez les paramètres de l'opérateur historique être l' [objet des paramètres de l'opérateur historique](#) .
2. Soit `script` actif le [script actif](#) .
3. Laissez le contexte d'exécution du script être nul.
4. Si le script actif n'est pas null, définissez le contexte d'exécution du script sur un nouveau [contexte d'exécution JavaScript](#) , avec son champ `Function` défini sur null , son champ `Realm` défini sur [le domaine](#) de l'objet de [paramètres](#) du script actif et son `ScriptOrModule` défini sur le script actif . [enregistrer](#) .

Comme on le voit ci-dessous, ceci est utilisé pour propager le [script actif](#) actuel jusqu'au moment où le rappel de travail est invoqué.

Un cas où le script actif n'est pas nul, et où il est utile de l'enregistrer de cette manière, est le suivant :

```
Promise.resolve('import(`./example.mjs`)').then(eval);
```

Sans cette étape (et les étapes qui l'utilisent dans [HostCallJobCallback](#) ), il n'y aurait pas [de script actif](#) lorsque l' `import()` expression est évaluée, puisqu'il `eval()` s'agit d'une fonction intégrée qui ne provient d'aucun [script](#) particulier .

Une fois cette étape en place, le script actif est propagé à partir du code ci-dessus dans le travail, ce qui permet `import()` d'utiliser l' [URL de base](#) du script d'origine de manière appropriée.



*active script* peut être nul si l'utilisateur clique sur le bouton suivant :

```
<button
onclick="Promise.resolve('import(`./example.mjs`)').then(eval
)">Click me</button>
```

Dans ce cas, la fonction JavaScript pour le [gestionnaire d'événements](#) sera créée par l' [obtention de la valeur actuelle de l'](#) algorithme du gestionnaire d'événements, qui crée une fonction avec une valeur `[[ScriptOrModule]]` nulle. Ainsi, lorsque la machinerie de promesse appelle [HostMakeJobCallback](#) , il n'y aura pas [de script actif](#) à transmettre.

Par conséquent, cela signifie que lorsque l' `import()` expression est évaluée, il n'y aura toujours pas [de script actif](#) . Heureusement, cela est géré par notre implémentation de [HostLoadImportedModule](#) en revenant à l'utilisation de [l'URL de base de l'API](#) de [l'objet de paramètres actuel](#) .

5. Renvoie l' [enregistrement JobCallback](#) { `[[Callback]]` : *callable* , `[[HostDefined]]` : { `[[IncumbentSettings]]` : *paramètres du titulaire* , `[[ActiveScriptContext]]` : *contexte d'exécution du script* } }.

#### 8.1.6.5 Crochets d'hôte liés au module

La spécification JavaScript définit une syntaxe pour les modules, ainsi que certaines parties indépendantes de l'hôte de leur modèle de traitement. Cette spécification définit le reste de leur modèle de traitement : comment le système de modules est amorcé, via l' `script` élément dont `type` l'attribut est défini sur " `module` ", et comment les modules sont récupérés, résolus et exécutés. [\[JAVASCRIPT\]](#)

*Bien que la spécification JavaScript parle en termes de « `scripts` » par rapport à « `modules` », en général, cette spécification parle en termes de [scripts classiques](#) par rapport aux [scripts de module](#) , puisque les deux utilisent l' `script` élément.*

**`modulePromise = import(specifieur)`**

Renvoie une promesse pour l'objet d'espace de noms de module pour le [script de module](#) identifié par *specifieur* . Cela permet l'importation dynamique des scripts de module lors de l'exécution, au lieu d'utiliser statiquement le `import` formulaire d'instruction. Le spécificateur sera [résolu](#) par rapport au [script actif](#) .

La promesse renvoyée sera rejetée si un spécificateur non valide est donné ou si un échec survient lors de [la récupération](#) ou de l'évaluation du graphe de module résultant.

Cette syntaxe peut être utilisée à la fois dans [les scripts classiques](#) et de module . Il fournit ainsi une passerelle vers le monde des scripts de modules, à partir du monde des scripts classiques.

**`url = import.meta.url`**

Renvoie l' [URL de base](#) du [script de module actif](#) .

Cette syntaxe ne peut être utilisée qu'à l'intérieur [des scripts de module](#) .

```
url = import.meta.resolve(specifier)
```

Renvoie *le spécificateur* , [résolu](#) par rapport au [script actif](#) . Autrement dit, cela renvoie l'URL qui serait importée à l'aide de `.import(specifier)`

Lève une `TypeError` exception si un spécificateur non valide est donné.

Cette syntaxe ne peut être utilisée qu'à l'intérieur [des scripts de module](#) .

Une **carte de module** est une [carte](#) indexée par [des tuples](#) consistant en un [enregistrement d'URL](#) et une [chaîne](#) . L' [enregistrement d'URL](#) est l' [URL de requête](#) à laquelle le module a été récupéré, et la [chaîne](#) indique le type du module (par exemple " `javascript` "). Les valeurs de [la carte de module](#) sont soit un [script de module](#) , null (utilisé pour représenter les échecs de récupération), soit une valeur d'espace réservé " `fetching` ". [Les cartes de module](#) sont utilisées pour garantir que les scripts de module importés ne sont récupérés, analysés et évalués qu'une seule fois par [Document](#) ou [worker](#) .

Étant donné que [les mappages de modules](#) sont indexés par (URL, type de module), le code suivant créera trois entrées distinctes dans le [mappage de module](#) , car il en résulte trois [tuples](#) différents (URL, type de module) (tous avec `javascript` le type " ") :

```
import "https://example.com/module.mjs";
import "https://example.com/module.mjs#map-buster";
import "https://example.com/module.mjs?debug=true";
```

C'est-à-dire que [les requêtes](#) et [les fragments](#) d'URL peuvent être modifiés pour créer des entrées distinctes dans la [carte du module](#) ; ils ne sont pas ignorés. Ainsi, trois extractions distinctes et trois évaluations de module distinctes seront effectuées.

En revanche, le code suivant ne créerait qu'une seule entrée dans le [module map](#) , car après avoir appliqué l' [analyseur d'URL à ces entrées, les enregistrements d'URL](#) résultants sont égaux :

```
import "https://example.com/module2.mjs";
import "https:example.com/module2.mjs";
import "https://example.com/module2.mjs";
import "https://example.com/foo/..module2.mjs";
```

Ainsi, dans ce deuxième exemple, une seule extraction et une seule évaluation de module auront lieu.

Notez que ce comportement est le même que la façon dont [les travailleurs partagés](#) sont indexés par leur [url de constructeur](#) analysé .

Étant donné que le type de module fait également partie de la clé [de mappage de module](#) , le code suivant créera deux entrées distinctes dans le [mappage de module](#) (le type est " `javascript` " pour le premier et " `css` " pour le second) :



```

<script type=module>
  import "https://example.com/module";
</script>
<script type=module>
  import "https://example.com/module" assert { type: "css" };
</script>

```

Cela peut entraîner l'exécution de deux extractions distinctes et de deux évaluations de module distinctes. Il s'agit d'une [violation délibérée](#) d'une contrainte recommandée (mais pas obligatoire) par la spécification des assertions d'importation indiquant que chaque appel à [HostLoadImportedModule](#) avec la même paire ( *referrer* , *moduleRequest* .[[*Spécifier*]]) doit renvoyer le même [Module Record](#) . [\[JSIMPORTASSERTIONS\]](#)

En pratique, en raison du cache mémoire non encore spécifié (voir le problème [#6110](#) ), la ressource ne peut être récupérée qu'une seule fois dans les navigateurs basés sur WebKit et Blink. De plus, tant que tous les types de module sont mutuellement exclusifs, la vérification du type de module lors de la [récupération d'un seul script de module](#) échouera pour au moins une des importations, de sorte qu'au plus une évaluation de module aura lieu.

Le but d'inclure le type dans la clé [de mappage de module](#) est de sorte qu'une importation avec la mauvaise assertion de type n'empêche pas une importation différente du même spécificateur mais avec le type correct de réussir.

Les scripts de module JavaScript sont le type d'importation par défaut lors de l'importation à partir d'un autre module JavaScript ; c'est-à-dire que lorsqu'une `import` instruction n'a pas `type` d'assertion d'importation, le type du script de module importé sera JavaScript. Toute tentative d'importation d'une ressource JavaScript à l'aide d'une `import` instruction avec une `type` assertion d'importation échouera :

```

<script type="module">
  // All of the following will fail, assuming that the imported
  // .mjs files are served with a
  // JavaScript MIME type. JavaScript module scripts are the
  // default and cannot be imported with
  // any import type assertion.
  import foo from "./foo.mjs" assert { type: "javascript" };
  import foo2 from "./foo2.mjs" assert { type: "js" };
  import foo3 from "./foo3.mjs" assert { type: "" };
  await import("./foo4.mjs", { assert: { type: null } });
  await import("./foo5.mjs", { assert: { type: undefined } });
</script>

```

#### 8.1.6.5.1 *HostGetImportMetaProperties ( moduleRecord )*



JavaScript contient une opération abstraite [HostGetImportMetaProperties définie par l'implémentation](#) . Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

1. Soit *moduleScript* la valeur *moduleRecord* .[[HostDefined]].
2. [Assert](#) : l'[URL de base](#) de *moduleScript* n'est pas nulle, car *moduleScript* est un [script de module JavaScript](#) .
3. Soit *urlString* l' [URL de base](#) de *moduleScript* , [sérialisée](#) .
4. Soit *étapes* les étapes suivantes, étant donné le *spécificateur* d'argument :
  1. Définissez le *spécificateur* sur ? [ToString](#) ( *spécificateur* ).
  2. Soit *url* le résultat de [la résolution d'un spécificateur de module](#) donné *moduleScript* et *spécifier* .
  3. Renvoie la [sérialisation](#) de *url* .
5. Soit *resolveFunction* ! [CreateBuiltinFunction](#) ( *étapes* , 1, " *resolve*", « » ).
6. Renvoie « [Enregistrer](#) { [[Clé]] : " *url*", [[Valeur]] : *urlString* }, [Enregistrer](#) { [[Clé]] : " *resolve* ", [[Valeur]] : *resolveFunction* } ».

#### 8.1.6.5.2 *HostGetSupportedImportAssertions ()*

La proposition *Import Assertions* contient une opération abstraite [HostGetSupportedImportAssertions définie par l'implémentation](#) . Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JSIMPORTASSERTIONS\]](#)

1. Retour « " *type* " ».

#### 8.1.6.5.3 *HostLoadImportedModule ( referrer , moduleRequest , loadState , payload )*

JavaScript contient une opération abstraite [HostLoadImportedModule définie par l'implémentation](#) . Les agents utilisateurs doivent utiliser l'implémentation suivante : [\[JAVASCRIPT\]](#)

Cette spécification s'attend à ce que le deuxième paramètre soit un [ModuleRequest Record](#) , au lieu d'une chaîne comme spécifié par ECMA-262. C'est sous l'hypothèse que la proposition d'assertions d'importation , lorsqu'elle est mise à jour pour utiliser [HostLoadImportedModule](#) au lieu des crochets de chargement de module précédents, mettra à jour l'opération abstraite en passant un [ModuleRequest Record](#) . [\[JSIMPORTASSERTIONS\]](#)

1. Soit `settingsObject` l' [objet de paramètres actuel](#) .
2. Si l'[objet global](#) de `settingsObject` implémente `or` et que `loadState` n'est pas défini, alors : [WorkletGlobalScopeServiceWorkerGlobalScope](#)

`loadState` est indéfini lorsque le processus de récupération en cours a été lancé par un `import()` appel dynamique, soit directement, soit lors du chargement des dépendances transitives du module importé dynamiquement.

1. Soit l'achèvement être [Completion Record](#) { `[[Type]]` : lancer, `[[Valeur]]` : un nouveau `TypeError`, `[[Cible]]` : vide }.
2. Effectuez [FinishLoadingImportedModule](#) ( `réfèrent` , `moduleRequest` , `charge utile` , `achèvement` ).
3. Retour.
3. Laissez `referencingScript` être null.
4. Laissez `fetchOptions` être les [options de récupération de script classiques par défaut](#) .
5. Soit `fetchReferrer` la valeur " `client` ".
6. Si le `réfèrent` est un [enregistrement de script](#) ou un [enregistrement de module](#) , alors :
  1. Définissez `referencingScript` sur `referrer` .`[[HostDefined]]`.
  2. Définissez `settingsObject` sur l'[objet settings](#) de `referencingScript` .
  3. Définissez `fetchOptions` sur les [options d'extraction du script descendant](#) pour [les options d'extraction](#) de `referencingScript` .
  4. [Assert](#) : `fetchOptions` n'est pas null, car `referencingScript` est un [script classique](#) ou un [script de module JavaScript](#) .
  5. Si aucune des conditions suivantes n'est vraie :
    - le `réfèrent` est un [enregistrement de script](#) ; ou
    - `referrer` est un [enregistrement de module](#) et `referrer` .`[[Status]]` est l'un des suivants : évaluation, évaluation-asynchrone ou évaluation,puis définissez `fetchReferrer` sur l'[URL de base](#) du `réfèrent` .

Nous définissons *fetchReferrer* de manière conditionnelle pour ne pas propager le référent lors de l'utilisation de `import()`. [Le numéro 3744](#) examine l'alignement des importations dynamiques avec les importations statiques.

7. *referrer* est généralement un [Script Record](#) ou un [Module Record](#), mais ce ne sera pas le cas pour les gestionnaires d'événements pour [obtenir la valeur actuelle de l'](#) algorithme du gestionnaire d'événements. Par exemple, étant donné :

```
8. <button onclick="import('./foo.mjs')">Click me</button>
```

9. Si un [click](#) événement se produit, alors au moment où l' `import()` expression s'exécute, [GetActiveScriptOrModule](#) renverra null et cette opération recevra le [domaine actuel](#) en tant que *réfèrent* de secours .
10. [Interdire les autres cartes d'importation](#) étant donné *settingsObject* .
11. Soit *url* le résultat de [la résolution d'un spécificateur de module](#) donné *referencingScript* et *moduleRequest* .[[*Spécifier*]], interceptant toutes les exceptions. S'ils lèvent une exception, laissez *resolutionError* être l'exception levée.
12. Si l'étape précédente a généré une exception, alors :
1. Soit l'achèvement être [Completion Record](#) { [[*Type*]] : throw, [[*Value*]] : *resolutionError*, [[*Target*]] : empty }.
  2. Effectuez [FinishLoadingImportedModule](#) ( *réfèrent* , *moduleRequest* , *c* *harge utile* , *achèvement* ).
  3. Retour.
13. Que *la destination* soit "script".
14. Si *loadState* n'est pas indéfini, définissez *la destination* sur *loadState* .[[*Destination*]].
15. [Récupérez un seul script de module importé](#) avec *url* , *settings object* , *destination* , *fetchOptions* , *fetchReferrer* , *moduleRequest* et *onSingle FetchComplete* comme défini ci-dessous. Si *loadState* n'est pas indéfini et que *loadState* .[[*PerformFetch*]] n'est pas nul, transmettez également *loadState* .[[*PerformFetch*]].

*onSingleFetchComplete moduleScript* donné est l'algorithme suivant :

1. Soit *la complétion* nulle.
2. Si *moduleScript* est nul, définissez l'achèvement sur [Completion Record](#) { [[*Type*]] : throw, [[*Value*]] : a new [TypeError](#), [[*Target*]] : empty }.

3. Sinon, si [l'erreur d'analyse](#) de *moduleScript* n'est pas nulle, alors :
  - Soit *parseError* l' [erreur d'analyse](#) de *moduleScript* .
  - Définissez la complétion sur [Completion Record](#) { *[[Type]]* : *throw*, *[[Value]]* : *parseError* , *[[Target]]* : *empty* }.
  - Si *loadState* n'est pas indéfini et que *loadState* .*[[ParseError]]* est nul, définissez *loadState* .*[[ParseError]]* sur *parseError* .
4. Sinon, définissez l'achèvement sur [Completion Record](#) { *[[Type]]* : *normal*, *[[Value]]* : *result* 's [record](#) , *[[Target]]* : *empty* }.
5. Effectuez [FinishLoadingImportedModule](#) ( *réfèrent* , *moduleRequest* , *c* *harge utile* , *achèvement* ).

## 8.1.7 Boucles d'événements

### 8.1.7.1 Définitions

Pour coordonner les événements, l'interaction de l'utilisateur, les scripts, le rendu, la mise en réseau, etc., les agents utilisateurs doivent utiliser **des boucles d'événements** comme décrit dans cette section. Chaque [agent](#) a une **boucle d'événements** associée , qui est unique à cet agent.

La [boucle d'événements](#) d'un [agent de fenêtre d'origine similaire](#) est connue sous le nom de **boucle d'événements de fenêtre** . La [boucle d'événements](#) d'un [agent de travail dédié](#) , [d'un agent de travail partagé](#) ou [d'un agent de travail de service](#) est appelée **boucle d'événement de travail** . Et la [boucle d'événements](#) d'un [agent de worklet](#) est connue sous le nom de **boucle d'événements de worklet** .

*[Les boucles d'événements](#) ne correspondent pas nécessairement aux threads d'implémentation. Par exemple, plusieurs [boucles d'événements de fenêtre](#) peuvent être planifiées de manière coopérative dans un seul thread.*

*Cependant, pour les différents [agents](#) de travail qui sont alloués avec *[[CanBlock]]* défini sur *true*, la spécification JavaScript leur impose des exigences concernant [forward progress](#) , ce qui revient en fait à exiger des threads dédiés par agent dans ces cas.*

---

Une [boucle d'événements](#) comporte une ou plusieurs **files d'attente de tâches** . Une [file d'attente de tâches](#) est un [ensemble](#) de [tâches](#) .

*Les files d'attente de tâches sont des ensembles , et non des files d'attente , car le modèle de traitement de la boucle d'événements récupère la première tâche exécutable de la file d'attente choisie, au lieu de retirer la première tâche de la file d'attente.*

*La file d'attente de microtâches n'est pas une file d'attente de tâches .*

Les tâches encapsulent des algorithmes qui sont responsables de travaux tels que :

### Événements

La répartition d'un Event objet sur un EventTarget objet particulier est souvent effectuée par une tâche dédiée.

*Tous les événements ne sont pas distribués à l'aide de la file d'attente des tâches ; beaucoup sont envoyés pendant d'autres tâches.*

### Analyse

L' analyseur HTML tokenisant un ou plusieurs octets, puis traitant les jetons résultants, est généralement une tâche.

### Rappels

L'appel d'un rappel est souvent effectué par une tâche dédiée.

### Utiliser une ressource

Lorsqu'un algorithme extrait une ressource, si l'extraction se produit de manière non bloquante, le traitement de la ressource une fois qu'une partie ou la totalité de la ressource est disponible est effectué par une tâche.

### Réagir à la manipulation du DOM

Certains éléments ont des tâches qui se déclenchent en réponse à une manipulation du DOM, par exemple lorsque cet élément est inséré dans le document .

Formellement, une **tâche** est une structure qui a :

#### ***Pas***

Une série d'étapes spécifiant le travail à effectuer par la tâche.

#### **Une source**

L'une des sources de tâches , utilisée pour regrouper et sérialiser les tâches associées.

#### **Un document**

A Document associé à la tâche, ou null pour les tâches qui ne sont pas dans une boucle d'événement de fenêtre .

#### **Un ensemble d'objets de paramètres d'environnement d'évaluation de script**

Ensemble d' objets de paramètres d'environnement utilisés pour suivre l'évaluation du script pendant la tâche .

Une [tâche](#) est **exécutable** si son [document](#) est null ou [entièrement actif](#) .

Selon son champ [source](#) , chaque [tâche](#) est définie comme provenant d'une **source de tâche** spécifique . Pour chaque [boucle d'événements](#) , chaque [source de tâche](#) doit être associée à une [file d'attente de tâches](#) spécifique .

*Essentiellement, [les sources de tâches](#) sont utilisées dans les normes pour séparer des types de tâches logiquement différents, qu'un agent utilisateur pourrait souhaiter distinguer. [Les files d'attente de tâches](#) sont utilisées par les agents utilisateurs pour regrouper les sources de tâches dans une [boucle d'événements](#) donnée .*

Par exemple, un agent utilisateur pourrait avoir une [file d'attente de tâches](#) pour les événements de souris et de touche (à laquelle la [source de tâche d'interaction de l'utilisateur](#) est associée) et une autre à laquelle toutes les autres [sources de tâche](#) sont associées. Ensuite, en utilisant la liberté accordée à l'étape initiale du [modèle de traitement de la boucle d'événements](#) , il pourrait donner la préférence aux événements du clavier et de la souris par rapport aux autres tâches les trois quarts du temps, en gardant l'interface réactive mais sans affamer les autres files d'attente de tâches. Notez que dans cette configuration, le modèle de traitement impose toujours que l'agent utilisateur ne traite jamais les événements d'une [source de tâche](#) dans le désordre.

---

Chaque [boucle d'événements](#) a une **tâche en cours d'exécution** , qui est soit une [tâche](#) , soit nulle. Initialement, c'est nul. Il est utilisé pour gérer la réentrance.

Chaque [boucle d'événements](#) a une **file d'attente de microtâches** , qui est une [file d'attente](#) de [microtâches](#) , initialement vide. Une **microtâche** est une manière familière de se référer à une [tâche](#) qui a été créée via la [file d'attente d'un algorithme de microtâche](#) .

Chaque [boucle d'événement](#) a un booléen **effectuant un point de contrôle de microtâche** , qui est initialement faux. Il est utilisé pour empêcher l'invocation réentrante de l' algorithme [d'exécution d'un point de contrôle de microtâche](#) .

Chaque [boucle d'événement de fenêtre](#) a un [DOMHighResTimeStamp](#) **temps de dernière opportunité de rendu** , initialement défini sur zéro.

Chaque [boucle d'événements de fenêtre](#) a une [DOMHighResTimeStamp](#) **heure de début de dernière période d'inactivité** , initialement définie sur zéro.

Pour obtenir les **mêmes fenêtres de boucle** pour une [boucle de boucle d'événement de fenêtre](#) , renvoyez tous les objets dont [la boucle d'événement](#) de l' [agent concerné](#) est `loop.Window`



### 8.1.7.2 Mise en file d'attente des tâches

Pour **mettre une tâche en file d'attente** sur une source [de source de tâche](#) , qui exécute une série d' *étapes* étapes , en fonction éventuellement d'une *boucle d'événement* boucle d'événement et d'un *document* document :

1. Si *la boucle d'événement* n'a pas été donnée, définissez *la boucle d'événement* sur la [boucle d'événement implicite](#) .
2. Si *document* n'a pas été donné, définissez *document* sur le [document implicite](#) .
3. Soit *tâche* une nouvelle [tâche](#) .
4. Définissez [les étapes](#) de *la tâche* sur *étapes* .
5. Définissez [la source](#) de *la tâche* sur *source* .
6. Définissez [le document](#) de *la tâche* sur le *document* .
7. Définissez [l'ensemble d'objets de paramètres d'environnement d'évaluation de script](#) de *la tâche* sur un [ensemble](#) vide .
8. Soit *queue* la [file d'attente de tâches](#) à laquelle *la source* est associée sur la *boucle d'événements* .
9. [Ajouter](#) *la tâche* à *la file d'attente* .

**Ne pas transmettre une boucle d'événement et un document à la file d'attente d'une tâche signifie s'appuyer sur la boucle d'événement implicite ambiguë et mal spécifiée et les concepts de document implicites . Les auteurs de spécifications doivent soit toujours transmettre ces valeurs, soit utiliser les algorithmes wrapper pour mettre en file d'attente une tâche globale ou mettre en file d'attente une tâche d'élément à la place. L'utilisation des algorithmes wrapper est recommandée.**

Pour **mettre en file d'attente une tâche globale** sur une source [source de tâches](#) , avec un [objet global](#) *global* et une série d' *étapes* étapes :

1. Soit *boucle d'événement* la [boucle d'événement](#) de l' [agent global](#) concerné .
2. Soit *document* associé à *global* , si *global* est un objet ; [Document](#) sinon nul. [Window](#)
3. [Mettre en file d'attente une tâche](#) en fonction de *la source* , de *la boucle d'événement* , du *document* et des *étapes* .

Pour **mettre en file d'attente un élément task** sur une source [source de tâches](#) , avec un *élément* element et une série d' *étapes* étapes :

1. Soit *global* l' [objet global pertinent](#) de l'*élément* .



2. [Mettre en file d'attente une tâche globale](#) donnée *source* , *global* et *steps* .

Pour **mettre en file d'attente une microtâche** qui exécute une série d' *étapes* , éventuellement avec une *boucle d'événement* et un *document* :

1. Si la *boucle d'événement* n'a pas été donnée, définissez la *boucle d'événement* sur la [boucle d'événement implicite](#) .
2. Si *document* n'a pas été donné, définissez *document* sur le [document implicite](#) .
3. Soit *microtask* une nouvelle [tâche](#) .
4. Définissez [les étapes](#) de la *microtâche* sur *étapes* .
5. Définissez [la source](#) de la *microtâche* sur la **source de la tâche de la microtâche** .
6. Définissez [le document](#) de la *microtâche* sur *document* .
7. Définissez [l'ensemble d'objets de paramètres d'environnement d'évaluation de script](#) de la *microtâche* sur un [ensemble](#) vide .
8. [Mettre](#) la *microtâche* en file d'attente dans la file d'attente des [microtâches de la boucle d'événements](#) .

*Il est possible qu'une [microtâche](#) soit déplacée vers une [file d'attente de tâches normale](#) si, lors de son exécution initiale, elle [fait tourner la boucle d'événements](#) . C'est le seul cas dans lequel l' [ensemble d'objets de paramètres d'environnement d'évaluation source](#) , [document](#) et [script de la microtâche](#) sont consultés ; ils sont ignorés par l' [algorithme d'exécution d'un point de contrôle de microtâche](#) .*

La **boucle événementielle implicite** lors de la mise en file d'attente d'une tâche est celle qui peut être déduite du contexte de l'algorithme appelant. Ceci est généralement sans ambiguïté, car la plupart des algorithmes de spécification n'impliquent jamais qu'un seul [agent](#) (et donc une seule [boucle d'événement](#) ). L'exception concerne les algorithmes impliquant ou spécifiant une communication inter-agents (par exemple, entre une fenêtre et un travailleur) ; dans ces cas, le concept [implicite de boucle d'événements](#) ne doit pas être invoqué et les spécifications doivent explicitement fournir une [boucle d'événements](#) lors [de la mise en file d'attente d'une tâche](#) ou [d'une microtâche](#) .

Le **document implicite** lors de la mise en file d'attente d'une tâche sur une *boucle d' événements de boucle* d'événements est déterminé comme suit :

1. Si la *boucle d'événements* n'est pas une [boucle d'événements de fenêtre](#) , renvoie null.

2. Si la tâche est mise en file d'attente dans le contexte d'un élément, renvoie le [nœud document](#) de l'élément .
3. Si la tâche est mise en file d'attente dans le contexte d'un [contexte de navigation](#) , renvoie le [document actif](#) du contexte de navigation .
4. Si la tâche est mise en file d'attente par ou pour un [script](#) , renvoyez les [paramètres](#) de l' [objet global](#) de l'objet [associéDocument](#) au script .
5. [Assert](#) : cette étape n'est jamais atteinte, car l'une des conditions précédentes est vraie. **Vraiment?**

La [boucle d'événement implicite](#) et le [document implicite](#) sont vaguement définis et ont beaucoup d'action à distance. L'espoir est de supprimer ces [documents, en particulier sous-entendus](#) . Voir [numéro 4980](#) .

### 8.1.7.3 Modèle de traitement

Une [boucle d'événements](#) doit continuellement parcourir les étapes suivantes tant qu'elle existe :

1. Laissez *la plus ancienneTask* et *taskStartTime* être nulles.
2. Si la [boucle d'événements](#) a une [file d'attente de tâches](#) avec au moins une [tâche exécutable](#) , alors :
  1. Soit *taskQueue* une telle [file d'attente de tâches](#) , choisie d'une manière [définie par l'implémentation](#) .

*N'oubliez pas que la [file d'attente de microtâches](#) n'est pas une [file d'attente de tâches](#) , elle ne sera donc pas choisie à cette étape. Cependant, une [file d'attente de tâches](#) à laquelle la [source de tâche de microtâche](#) est associée peut être choisie à cette étape. Dans ce cas, la [tâche](#) choisie à l'étape suivante était à l'origine une [microtâche](#) , mais elle a été déplacée dans le cadre de [la rotation de la boucle d'événements](#) .*

2. Définissez *taskStartTime* sur l' [heure actuelle partagée non sécurisée](#) .
3. Définissez *ancientTask* sur la première [tâche exécutable](#) dans *taskQueue* et [supprimez](#) -la de *taskQueue* .
4. Définissez la [tâche en cours d'exécution](#) de [la boucle d'événements](#) sur *ancientTask* .
5. Effectuez [les étapes](#) de *la plus ancienne tâche* .

6. Redéfinissez la tâche en cours d'exécution de la boucle d'événements sur null.
3. Effectuez un point de contrôle de microtâche .
4. Soit *hasARenderingOpportunity* faux.
5. Soit *maintenant* l' heure actuelle partagée non sécurisée . [THS]
6. Si *la tâche la plus ancienne* n'est pas nulle, alors :
  1. Soit *les contextes de navigation de niveau supérieur* un ensemble vide .
  2. Pour chaque paramètre d'objet de paramètres d'environnement de l'ensemble d'objets de paramètres d'environnement d'évaluation de script de *ancientTask* :
    1. Soit *global* l' objet global de *settings* .
    2. Si *global* n'est pas un Window objet, continuez .
    3. Si le contexte de navigation de *global* est nul, alors continuez .
    4. Soit *tlbc* le contexte de navigation de niveau supérieur du contexte de navigation *global* .
    5. Si *tlbc* n'est pas nul, ajoutez -le aux *contextes de navigation de niveau supérieur* .
  3. Signalez les tâches longues en transmettant *taskStartTime* , *maintenant* (l'heure de fin de la tâche), *les contextes de navigation de niveau supérieur* et *la plus ancienneTask* .
7. **Mettez à jour le rendu** : s'il s'agit d'une boucle d'événement de fenêtre , alors :
  1. Soit *docs* tous Document les objets dont la boucle d'événements de l'agent concerné est cette boucle d'événements, triés arbitrairement sauf que les conditions suivantes doivent être remplies :
    1. Tout Document *B* dont le document conteneur est *A* doit figurer après *A* dans la liste.
    2. S'il y a deux documents *A* et *B* qui ont tous deux le même document conteneur non nul *C* , alors l'ordre de *A* et *B* dans la liste doit correspondre à l' ordre de l'arborescence shadow-inclusive de leurs conteneurs navigables respectifs dans l' arborescence de nœuds de *C* .

Dans les étapes ci-dessous qui itèrent sur *docs* , chacun Document doit être traité dans l'ordre dans lequel il se trouve dans la liste.

2. *Opportunités de rendu* : Supprimez des docs tous Document les objets dont les nœuds navigables n'ont pas d' opportunité de rendu .

Un navigable a une **opportunité de rendu** si l'agent utilisateur est actuellement en mesure de présenter le contenu du navigable à l'utilisateur, en tenant compte des contraintes de taux de rafraîchissement matériel et de la limitation de l'agent utilisateur pour des raisons de performances, mais en considérant le contenu présentable même s'il se trouve en dehors de la fenêtre d'affichage.

Un navigable n'a aucune opportunité de rendu si son document actif est rendu bloqué ; sinon, les opportunités de rendu sont déterminées en fonction des contraintes matérielles telles que les taux de rafraîchissement de l'affichage et d'autres facteurs tels que les performances de la page ou si l' état de visibilité du document est " visible". Les opportunités de rendu se produisent généralement à intervalles réguliers.

*Cette spécification n'impose aucun modèle particulier pour la sélection des opportunités de rendu. Mais par exemple, si le navigateur tente d'atteindre un taux de rafraîchissement de 60 Hz, les opportunités de rendu se produisent au maximum toutes les 60 secondes (environ 16,7 ms). Si le navigateur constate qu'un navigable n'est pas en mesure de maintenir ce taux, il peut tomber à 30 opportunités de rendu par seconde plus durables pour ce navigable , plutôt que de perdre occasionnellement des images. De même, si un navigable n'est pas visible, l'agent utilisateur peut décider de déposer cette page à un rendu beaucoup plus lent de 4 opportunités par seconde, voire moins.*

3. Si docs n'est pas vide, définissez `hasARenderingOpportunity` sur true et définissez l'heure de la dernière opportunité de rendu de cette boucle d' événements sur `taskStartTime` .
4. *Rendu inutile* : supprimez des documents tous Document les objets qui remplissent les deux conditions suivantes :
  1. l'agent utilisateur pense que la mise à jour du rendu du Document nœud navigable de n'aurait aucun effet visible, et
  2. la Document carte des rappels d'images d'animation est vide.
5. Supprimez de la documentation tous Document les objets pour lesquels l'agent utilisateur estime qu'il est préférable d'ignorer la mise à jour du rendu pour d'autres raisons.

*L'étape intitulée Opportunités de rendu empêche l'agent utilisateur de mettre à jour le rendu lorsqu'il est incapable de présenter un nouveau contenu à l'utilisateur (il n'y a pas d' opportunité de rendu ).*

*L'étape intitulée Rendu inutile empêche l'agent utilisateur de mettre à jour le rendu lorsqu'il n'y a pas de nouveau contenu à dessiner.*

Cette étape permet à l'agent utilisateur d'empêcher les étapes ci-dessous de s'exécuter pour d'autres raisons, par exemple, pour s'assurer que certaines tâches sont exécutées immédiatement les unes après les autres, avec seulement des points de contrôle de microtâche entrelacés (et sans, par exemple, des rappels de trame d'animation entrelacés). Concrètement, un agent utilisateur peut souhaiter fusionner les rappels de minuterie ensemble, sans mises à jour de rendu intermédiaires.

6. Pour chaque entièrement actif Document dans *docs* , videz les candidats de mise au point automatique pour cela Document si son nœud navigable est un traversable de niveau supérieur .
7. Pour chaque entièrement actif Document dans *les documents* , exécutez les étapes de redimensionnement pour cela Document. [VUE CSSOM]
8. Pour chaque entièrement actif Document dans *les docs* , exécutez les étapes de défilement pour cela Document. [VUE CSSOM]
9. Pour chaque document pleinement actif , évaluez les requêtes multimédias et signalez les modifications correspondantes . [VUE CSSOM]DocumentDocument
10. Pour chaque entièrement actif Document dans *docs* , mettez à jour les animations et envoyez des événements pour cela Document, en passant *maintenant* comme horodatage. [WEBANIMATIONS]
11. Pour chaque entièrement actif Document dans *les docs* , exécutez les étapes plein écran pour cela Document. [PLEIN ÉCRAN]
12. Pour chaque entièrement actif Document dans *docs* , si l'agent utilisateur détecte que le stockage de sauvegarde associé à un CanvasRenderingContext2D ou un OffscreenCanvasRenderingContext2D, *context* , a été perdu, alors il doit exécuter les **étapes context lost** pour chacun de ces *contextes* :
  1. Soit *canvas* la valeur de l'attribut *context*canvas , si *context* est a CanvasRenderingContext2D, ou l' objet associéOffscreenCanvas pour *context* sinon.
  2. Définissez le *contexte* du contexte perdu sur vrai.
  3. Réinitialisez le contexte de rendu à son état par défaut étant donné *context* .
  4. Soit *shouldRestore* le résultat du déclenchement d'un événement nommé contextlost at *canvas* , avec l' cancelable attribut initialisé à true.
  5. Si *shouldRestore* est faux, abandonnez ces étapes.

6. Essayez de restaurer *le contexte* en créant un stockage de sauvegarde à l'aide des attributs du *contexte* et en les associant au *contexte* . Si cela échoue, abandonnez ces étapes.
  7. Définissez *le contexte* perdu du contexte sur faux.
  8. Lancez un événement nommé contextrestored sur *canvas* .
13. Pour chaque entièrement actif Document dans *docs* , exécutez les rappels de trame d'animation pour cela Document, en passant *maintenant* comme horodatage.
14. Pour chaque *document* entièrement actif dans *docs* : Document
1. Recalculer les styles et mettre à jour la mise en page pour *doc* .
  2. Soit *la profondeur* égale à 0.
  3. Rassemblez les observations de redimensionnement actives en profondeur pour *doc* .
  4. Alors que *doc* a des observations de redimensionnement actives :
    1. Définissez *la profondeur* sur le résultat de la diffusion des observations de redimensionnement actives données *doc* .
    2. Recalculer les styles et mettre à jour la mise en page pour *doc* .
    3. Rassemblez les observations de redimensionnement actives en profondeur pour *doc* .
  5. Si *doc* a sauté les observations de redimensionnement , alors fournir une erreur de boucle de redimensionnement donnée *doc* .
15. Pour chaque entièrement actif Document dans *les docs* , si la zone ciblée n'est Document pas une zone focalisable , exécutez les étapes de mise au point pour Document cette fenêtre .

Par exemple, cela peut se produire parce qu'un élément a l' hidden attribut ajouté, provoquant l'arrêt de son rendu . Cela peut également arriver à un input élément lorsque l'élément est désactivé . *Cela déclenchera généralement* blur *des événements, et éventuellement* change *des événements.* *En plus de cette correction asynchrone, si la* zone ciblée du document *est supprimée, il y a une* correction synchrone . *Celui-là ne fera pas feu* blur *ni* change *événements.*



16. Pour chaque entièrement actif Document dans *docs* , exécutez les étapes de mise à jour des observations d'intersection pour cela Document, en passant *maintenant* comme horodatage. [OBSERVATEUR D'INTERSECTION]
17. Invoquez l' algorithme de synchronisation de la peinture de marque pour chaque Document objet dans *docs* .
18. Pour chaque entièrement actif Document dans *docs* , mettez à jour le rendu ou l'interface utilisateur de celui-ci Document et son nœud navigable pour refléter l'état actuel.

8. Si tous les éléments suivants sont vrais

1. ceci est une boucle d'événement de fenêtre
2. il n'y a pas de tâche dans les files d'attente de tâches de cette boucle d'événements dont le document est entièrement actif
3. la file d'attente des microtâches de cette boucle d'événements est vide
4. *hasARenderingOpportunity* est faux

alors:

5. Définissez l' heure de début de la dernière période d' inactivité de cette boucle d' événements sur l' heure actuelle partagée non sécurisée .
6. Soit *computeDeadline* les étapes suivantes :

1. Soit *la date limite* soit l'heure de début de la dernière période d'inactivité de cette boucle d'événements plus 50.

*Le plafond de 50 ms à l'avenir est d'assurer la réactivité aux nouvelles entrées de l'utilisateur dans le seuil de perception humaine.*

2. Soit *hasPendingRenders* faux.
3. Pour chaque *windowInSameLoop* des fenêtres de même boucle pour cette boucle d'événements :
  1. Si la carte des rappels d'images d'animation de *windowInSameLoop* n'est pas vide ou si l'agent utilisateur pense que *windowInSameLoop* peut avoir des mises à jour de rendu en attente, définissez *hasPendingRenders* sur true.
  2. Laissez *timerCallbackEstimates* être le résultat de l'obtention des valeurs de la carte des minuteurs actifs de *windowInSameLoop* .
  3. Pour chaque *timeoutDeadline* de *timerCallbackEstimates* ,

si *timeoutDeadline* est inférieur à *date limite* ,  
définissez *date limite* sur *timeoutDeadline* .

4. Si *hasPendingRenders* est vrai, alors :

1. Soit *nextRenderDeadline* [la dernière opportunité de rendu](#) de cette [boucle d'événements](#) plus (1000 divisé par le taux de rafraîchissement actuel).

Le taux de rafraîchissement peut être spécifique au matériel ou à l'implémentation. Pour un taux de rafraîchissement de 60 Hz, le *nextRenderDeadline* serait d'environ 16,67 ms après la [dernière opportunité de rendu](#) .

2. Si *nextRenderDeadline* est inférieur à *la date limite* , alors retournez *nextRenderDeadline* .

5. *Date limite* de retour .

7. Pour chaque victoire des [fenêtres de la même boucle](#) pour cette [boucle d'événements](#) , exécutez l' [algorithme de démarrage d'une période d'inactivité](#) pour la victoire avec l'étape suivante : retournez le résultat de l'appel de *computeDeadline* , [grossi](#) compte tenu [de la capacité d'isolement cross-origin](#) de l' [objet de paramètres pertinents de win](#) . [\[DEMANDE DE RAPPEL\]](#)

9. S'il s'agit d'une [boucle d'événements de travail](#) , alors :

1. Si l'[objet global](#) du [domaine](#) unique de l' [agent de cette boucle d'événements](#) est [pris en charge](#) et que l'agent utilisateur estime qu'il bénéficierait d'une mise à jour de son rendu à ce moment-là, alors : [DedicatedWorkerGlobalScope](#)
  1. Soit *maintenant* le [temps haute résolution actuel](#) étant donné le [DedicatedWorkerGlobalScope](#). [\[THS\]](#)
  2. [Exécutez les rappels de trame d'animation](#) pour cela [DedicatedWorkerGlobalScope](#), en passant *maintenant* comme horodatage.
  3. Mettez à jour le rendu de ce travailleur dédié pour refléter l'état actuel.

*Semblable aux notes de [mise à jour du rendu](#) dans une [boucle d'événement de fenêtre](#) , un agent utilisateur peut déterminer le taux de rendu dans le worker dédié.*

2. S'il n'y a pas [de tâches](#) dans les files [d'attente de tâches](#) de [la boucle d'événements](#) et que l'indicateur [de fermeture](#) de l'objet est vrai, alors détruisez la [boucle d'événements](#) , annulez ces étapes et reprenez les



étapes [d'exécution d'un agent](#) décrites dans la section [des agents Web ci-dessous](#).[WorkerGlobalScope](#)

---

Lorsqu'un agent utilisateur doit **effectuer un point de contrôle de microtâche** :

1. Si la boucle d' [événements effectuant un point de contrôle de microtâche](#) est vraie, alors retournez.
2. Définissez la boucle [d'événements effectuant un point de contrôle de microtâche](#) sur true.
3. Tant que la [file d'attente des microtâches](#) de [la boucle d'événements](#) n'est pas [vide](#) :
  1. Supposons que *la microtâche la plus ancienne* soit le résultat du [retrait de la file d'attente des microtâches](#) de la [boucle d'événements](#) .
  2. Définissez la [tâche en cours d'exécution](#) de [la boucle d'événements](#) sur *ancientMicrotask* .
  3. Exécutez *la plus ancienne Microtask* .

*Cela peut impliquer l'appel de rappels scriptés, qui appellent finalement le [nettoyage après l'exécution](#) des étapes de script, qui appellent cela [effectuer](#) à nouveau un algorithme de point de contrôle de microtâche, c'est pourquoi nous utilisons l' indicateur [d'exécution d'un point de contrôle de microtâche](#) pour éviter la réentrance.*

4. Redéfinissez la [tâche en cours d'exécution](#) de [la boucle d'événements](#) sur null.
4. Pour chaque [objet de paramètres d'environnement](#) dont [la boucle d'événements responsable](#) est cette [boucle d'événements](#) , [notifiez les promesses rejetées](#) sur cet [objet de paramètres d'environnement](#) .
5. [Nettoyez les transactions de la base de données indexée](#) .
6. Effectuez [ClearKeptObjects](#) ().

*Lorsque `WeakRef.prototype.deref()` renvoie un objet, cet objet est maintenu en vie jusqu'à la prochaine invocation de [ClearKeptObjects](#) (), après quoi il est à nouveau soumis à la récupération de place.*

7. Définissez la boucle [d'événements effectuant un point de contrôle de microtâche](#) sur false.

---

Lorsqu'un algorithme exécuté [en parallèle](#) doit **attendre un état stable**, l'agent utilisateur doit [mettre en file d'attente une microtâche](#) qui exécute les étapes suivantes, puis doit arrêter de s'exécuter (l'exécution de l'algorithme reprend lorsque la microtâche est exécutée, comme décrit dans les étapes suivantes) :

1. Exécutez la **section synchrone** de l'algorithme .
2. Reprend l'exécution de l'algorithme [en parallèle](#) , le cas échéant, comme décrit dans les étapes de l'algorithme.

*Les étapes des [sections synchrones](#) sont marquées d'un ⌚.*

---

Les étapes de l'algorithme qui disent de **faire tourner la boucle d'événements** jusqu'à ce qu'un *objectif* de condition soit atteint sont équivalentes à la substitution dans les étapes de l'algorithme suivantes :

1. Soit *tâche* [la tâche en cours d'exécution](#) de la [boucle d'événements](#) .

*tâche pourrait être une [microtâche](#) .*

2. Laissez *la source de la tâche* être [la source](#) de la *tâche* .
3. Soit *old stack* une copie de la [pile de contexte d'exécution JavaScript](#) .
4. Videz la [pile de contexte d'exécution JavaScript](#) .
5. [Effectuez un point de contrôle de microtâche](#) .

*Si la tâche est une [microtâche](#) , cette étape sera une non-opération car [l'exécution d'un point de contrôle de microtâche](#) est vraie.*

6. [En parallèle](#) :
  1. Attendez que l' *objectif* de la condition soit atteint.
  2. [Mettre une tâche](#) en file d'attente sur *la source de la tâche* pour :
    1. Remplacez la [pile de contexte d'exécution JavaScript](#) par *l'ancienne pile* .
    2. Effectuez toutes les étapes qui apparaissent après cette [rotation de l'](#) instance de boucle d'événements dans l'algorithme d'origine.

*Cela reprend la tâche .*

7. Arrêter *la tâche* , permettant à l'algorithme qui l'a invoqué de reprendre.

*Cela entraîne la poursuite de l'ensemble principal d'étapes de [la boucle d'événements](#) ou l' exécution d'un algorithme [de point de contrôle de microtâche](#) .*

*Contrairement à d'autres algorithmes de cette spécification et d'autres, qui se comportent de la même manière que les appels de fonction du langage de programmation, la [rotation de la boucle d'événements](#) ressemble plus à une macro, ce qui évite la saisie et l'indentation sur le site d'utilisation en se développant en une série d'étapes et d'opérations.*

Un algorithme dont les étapes sont :

1. Faire quelque chose.
2. [Faites tourner la boucle d'événements](#) jusqu'à ce que la génialité se produise.
3. Faites autre chose.

est un raccourci qui, après "développement macro", devient

1. Faire quelque chose.
2. Soit *old stack* une copie de la [pile de contexte d'exécution JavaScript](#) .
3. Videz la [pile de contexte d'exécution JavaScript](#) .
4. [Effectuez un point de contrôle de microtâche](#) .
5. [En parallèle](#) :
  1. Attendez que la génialité se produise.
  2. [Mettre en file d'attente une tâche](#) sur la source de tâche dans laquelle "faire quelque chose" a été fait pour :
    1. Remplacez la [pile de contexte d'exécution JavaScript](#) par *l'ancienne pile* .
    2. Faites autre chose.

Voici un exemple plus complet de la substitution, où la boucle d'événements est lancée depuis l'intérieur d'une tâche mise en file d'attente depuis le travail en parallèle. La version utilisant [spin the event loop](#) :

1. [En parallèle](#) :
  1. Faites la chose parallèle 1.
  2. [Mettez une tâche en file d'attente](#) sur la [source de la tâche de manipulation DOM](#) pour :

1. Faites la tâche 1.
2. [Faites tourner la boucle d'événements](#) jusqu'à ce que la génialité se produise.
3. Faites la tâche 2.

3. Faites la chose parallèle 2.

La version entièrement étendue :

1. [En parallèle](#) :

1. Faites la chose parallèle 1.
2. Laissez *l'ancienne pile* être nulle.
3. [Mettez une tâche en file d'attente](#) sur la [source de la tâche de manipulation DOM](#) pour :

1. Faites la tâche 1.
2. Définissez *l'ancienne pile* sur une copie de la [pile de contexte d'exécution JavaScript](#) .
3. Videz la [pile de contexte d'exécution JavaScript](#) .
4. [Effectuez un point de contrôle de microtâche](#) .

4. Attendez que la génialité se produise.
5. [Mettez une tâche en file d'attente](#) sur la [source de la tâche de manipulation DOM](#) pour :

1. Remplacez la [pile de contexte d'exécution JavaScript](#) par *l'ancienne pile* .
2. Faites la tâche 2.

6. Faites la chose parallèle 2.

---

Certains des algorithmes de cette spécification, pour des raisons historiques, exigent que l'agent utilisateur s'interrompe **pendant** l'exécution d'une [tâche](#) jusqu'à ce qu'un *objectif* de condition soit atteint. Cela signifie exécuter les étapes suivantes :

1. Si nécessaire, mettez à jour le rendu ou l'interface utilisateur de any [Document](#) ou [navigable](#) pour refléter l'état actuel.

2. Attendez que l' *objectif* de la condition soit atteint. Lorsqu'un agent utilisateur a une tâche en pause , la boucle d'événements correspondante ne doit pas exécuter d'autres tâches et tout script de la tâche en cours d'exécution doit se bloquer. Les agents utilisateurs doivent rester réactifs à l'entrée de l'utilisateur lorsqu'ils sont en pause, bien que dans une capacité réduite puisque la boucle d'événements ne fera rien.

***La pause est très préjudiciable à l'expérience utilisateur, en particulier dans les scénarios où une seule boucle d'événements est partagée entre plusieurs documents. Les agents utilisateurs sont encouragés à expérimenter des alternatives à la pause , comme faire tourner la boucle d'événements ou même simplement continuer sans aucune sorte d'exécution suspendue, dans la mesure où il est possible de le faire tout en préservant la compatibilité avec le contenu existant. Cette spécification changera volontiers si une alternative moins drastique est découverte comme étant compatible avec le Web.***

***Dans l'intervalle, les implémenteurs doivent être conscients que la variété d'alternatives que les agents utilisateurs pourraient expérimenter peut modifier des aspects subtils du comportement de la boucle d'événements , y compris la synchronisation des tâches et des microtâches . Les implémentations doivent continuer à expérimenter même si cela les amène à violer la sémantique exacte impliquée par l' opération de pause .***

#### 8.1.7.4 Sources de tâches génériques

Les sources de tâches suivantes sont utilisées par un certain nombre de fonctionnalités pour la plupart non liées dans cette spécification et d'autres.

##### **La source de la tâche de manipulation du DOM**

Cette source de tâches est utilisée pour les fonctionnalités qui réagissent aux manipulations du DOM, telles que les événements qui se produisent de manière non bloquante lorsqu'un élément est inséré dans le document .

##### **La source de tâche d'interaction utilisateur**

Cette source de tâches est utilisée pour les fonctionnalités qui réagissent à l'interaction de l'utilisateur, par exemple la saisie au clavier ou à la souris.

Les événements envoyés en réponse à l'entrée de l'utilisateur (par exemple, click les événements) doivent être déclenchés à l'aide de tâches mises en file d'attente avec la source de tâche d'interaction utilisateur . [UIÉVÉNEMENTS]

##### **La source de tâches réseau**

Cette source de tâche est utilisée pour les fonctionnalités qui se déclenchent en réponse à l'activité du réseau.

## La source des tâches de navigation et de parcours

Cette [source de tâches](#) est utilisée pour mettre en file d'attente les tâches impliquées dans [la navigation](#) et [la traversée de l'historique](#) .

### 8.1.7.5 Traitement de la boucle d'événements à partir d'autres spécifications

Écrire des spécifications qui interagissent correctement avec la [boucle d'événements](#) peut être délicat. Ceci est aggravé par la façon dont cette spécification utilise une terminologie indépendante du modèle de concurrence, nous disons donc des choses comme « [boucle d'événement](#) » et « [en parallèle](#) » au lieu d'utiliser des termes plus familiers spécifiques au modèle comme « thread principal » ou « sur un thread d'arrière-plan » . .

Par défaut, le texte de spécification s'exécute généralement sur la [boucle d'événements](#) . Cela découle du [modèle formel de traitement de la boucle d'événements](#) , en ce sens que vous pouvez éventuellement retracer la plupart des algorithmes jusqu'à une [tâche qui y est mise en file d'attente](#) .

Les étapes de l'algorithme de toute méthode JavaScript seront invoquées par le code de l'auteur appelant cette méthode. Et le code d'auteur ne peut être exécuté que via des tâches en file d'attente, provenant généralement de quelque part dans le [script modèle de traitement](#) .

À partir de ce point de départ, la directive primordiale est que tout travail qu'une spécification doit effectuer et qui bloquerait autrement la [boucle d'événements](#) doit plutôt être effectué [en parallèle](#) avec elle. Cela inclut (mais n'est pas limité à) :

- effectuer des calculs lourds ;
- afficher une invite destinée à l'utilisateur ;
- effectuer des opérations qui pourraient nécessiter l'intervention de systèmes extérieurs (c'est-à-dire "sortir du processus").

La complication suivante est que, dans les sections d'algorithme qui sont [en parallèle](#) , vous ne devez pas créer ou manipuler d'objets associés à un [objet spécifique de paramètres de domaine](#) , [global](#) ou d'environnement . (En termes plus familiers, vous ne devez pas accéder directement aux artefacts du thread principal à partir d'un thread d'arrière-plan.) Cela créerait des courses de données observables par le code JavaScript, car après tout, vos étapes d'algorithme s'exécutent en parallèle avec le code [JavaScript](#) .

Vous pouvez, cependant, manipuler les structures de données et les valeurs au niveau de la spécification à partir d' *Infra* , car celles-ci sont indépendantes du domaine. Ils ne sont jamais directement exposés à JavaScript sans qu'une conversion spécifique ait lieu (souvent [via Web IDL](#) ). [\[INFRA\]](#) [\[WEBIDL\]](#)

Pour affecter le monde des objets JavaScript observables, vous devez donc [mettre en file d'attente une tâche globale](#) pour effectuer de telles manipulations. Cela garantit que vos étapes sont correctement imbriquées par rapport aux autres événements qui se produisent dans la [boucle d'événements](#). De plus, vous devez choisir une [source de tâche](#) lors [de la mise en file d'attente d'une tâche globale](#) ; cela régit l'ordre relatif de vos pas par rapport aux autres. Si vous n'êtes pas sûr de [la source de tâche](#) à utiliser, choisissez l'une des [sources de tâches génériques](#) qui semble la plus applicable. Enfin, vous devez indiquer à quel [objet global](#) votre tâche en file d'attente est associée ; cela garantit que si cet objet global est inactif, la tâche ne s'exécute pas.

*La primitive de base, sur laquelle [se construit une tâche globale](#), est la [file d'attente d'un algorithme de tâche](#). En général, il est préférable [de mettre en file d'attente une tâche globale](#) car elle sélectionne automatiquement la bonne [boucle d'événements](#) et, le cas échéant, [document](#). Les spécifications plus anciennes utilisent souvent [la mise en file d'attente d'une tâche combinée à la boucle d'événements implicite](#) et aux concepts [de document implicites](#), mais cela est déconseillé.*

En mettant tout cela ensemble, nous pouvons fournir un modèle pour un algorithme typique qui doit fonctionner de manière asynchrone :

1. Effectuez tout travail de configuration synchrone, tout en restant sur la [boucle d'événements](#). Cela peut inclure la conversion de valeurs JavaScript spécifiques [au domaine](#) en valeurs de niveau de spécification indépendantes du domaine.
2. [Effectuez en parallèle](#) un ensemble d'étapes potentiellement coûteuses, fonctionnant entièrement sur des valeurs indépendantes du domaine et produisant un résultat indépendant du domaine.
3. [Mettez en file d'attente une tâche globale](#), sur une [source de tâche](#) spécifiée et avec un [objet global](#) approprié, pour reconvertir le résultat indépendant du domaine en effets observables sur le monde observable des objets JavaScript sur la [boucle d'événement](#).

[Voici un algorithme qui "chiffre" une liste](#) transmise de [chaînes de valeurs scalaires](#) *input*, après les avoir analysées en tant qu'URL :

1. Soit *urls* une [liste](#) vide.
2. [Pour chaque](#) *chaîne* d'entrée :
  1. Soit *parsed* le résultat de [l'analyse](#) de la *chaîne* par rapport à l' [objet de paramètres actuel](#).
  2. Si *l'analyse* est un échec, renvoie [une promesse rejetée](#) avec un `"SyntaxError"` [DOMException](#).
  3. Soit *sérialisé* le résultat de l'application du [séréaliseur d'URL](#) à *parsed*.



4. [Ajouter](#) *sérialisé* aux *URL* .
3. Soit *realm* le [domaine actuel](#) .
4. Soit *p* une nouvelle promesse.
5. Exécutez les étapes suivantes [en parallèle](#) :
  1. Laissez les *URL cryptées* être une [liste](#) vide .
  2. [Pour chaque](#) *url* d' *urls* :
    1. Attendez 100 millisecondes, pour que les gens pensent que nous faisons du cryptage intensif.
    2. Soit *crypté* une nouvelle [chaîne](#) dérivée de *url* , dont la *n* ième [unité de code](#) est égale à la *n* ième [unité de code](#) de *url* plus 13.
    3. [Ajouter](#) les *URL cryptées* aux *URL cryptées* .
  3. [Placez une tâche globale](#) en file d'attente sur la [source de tâche réseau](#) , en fonction de [l'objet global](#) du *domaine* , pour effectuer les étapes suivantes :
    1. Soit *array* le résultat de [la conversion](#) d'*URL cryptées* en un tableau JavaScript, dans *realm* .
    2. Résoudre *p* avec *array* .
6. Retour *p* .

Voici plusieurs choses à noter à propos de cet algorithme :

- Il effectue son analyse d'URL en amont, sur la [boucle d'événements](#) , avant de passer aux étapes [parallèles](#) . Cela est nécessaire, car l'analyse dépend de l' [objet de paramètres actuel](#) , qui ne serait plus actuel après être passé [en parallèle](#) .
- Alternativement, il aurait pu enregistrer une référence à [l'URL de base de l'API](#) de [l'objet de paramètres actuel](#) et l'utiliser lors des étapes [parallèles](#) ; cela aurait été équivalent. Cependant, nous vous recommandons plutôt de faire autant de travail que possible à l'avance, comme le fait cet exemple. Tenter d'enregistrer les valeurs correctes peut être source d'erreurs ; par exemple, si nous avions enregistré uniquement l' [objet de paramètres actuel](#) , au lieu de son [URL de base d'API](#) , il y aurait eu une course potentielle.
- Il passe implicitement une [liste](#) de [chaînes](#) des étapes initiales aux étapes [parallèles](#) . C'est OK, car [les listes](#) et [les chaînes](#) sont indépendantes [du domaine](#) .



- Il effectue un "calcul coûteux" (attendant 100 millisecondes par URL d'entrée) pendant les étapes [en parallèle , ne bloquant ainsi pas la boucle d'événement](#) principale .
- Les promesses, en tant qu'objets JavaScript observables, ne sont jamais créées et manipulées au cours des étapes [parallèles](#) . *p* est créé avant d'entrer dans ces étapes, puis est manipulé au cours d'une [tâche](#) mise [en file d'attente](#) spécifiquement à cette fin.
- La création d'un objet de tableau JavaScript se produit également pendant la tâche en file d'attente et veille à spécifier dans quel domaine il crée le tableau, car cela n'est plus évident d'après le contexte.

(Sur ces deux derniers points, voir aussi [whatwg/webidl issue #135](#) et [whatwg/webidl issue #371](#) , où nous réfléchissons encore aux subtilités du modèle de résolution de promesse ci-dessus.)

Une autre chose à noter est que, dans le cas où cet algorithme était appelé à partir d'une opération spécifiée par Web IDL prenant un `sequence< USVString>`, il y avait une conversion automatique des objets JavaScript spécifiques [au domaine](#) fournis par l'auteur en tant qu'entrée, dans le domaine indépendant `sequence< USVString>` Type Web IDL, que nous traitons ensuite comme une [liste](#) de [chaînes de valeurs scalaires](#) . Ainsi, selon la structure de votre spécification, il peut y avoir d'autres étapes implicites se produisant sur la [boucle d'événements](#) principale qui jouent un rôle dans tout ce processus pour vous préparer à aller [en parallèle](#) .

## 8.1.8 Événements

### 8.1.8.1 Gestionnaires d'événements

MDN

De nombreux objets peuvent avoir **des gestionnaires d'événements** spécifiés. [Ceux-ci agissent comme des écouteurs d'événements](#) sans capture pour l'objet sur lequel ils sont spécifiés. [\[DOM\]](#)

Un [gestionnaire d'événements](#) est une [structure](#) à deux [éléments](#) :

- une **valeur** , qui est soit null, un objet de rappel ou un [gestionnaire interne brut non compilé](#) . Le `EventHandler` type de fonction de rappel décrit comment cela est exposé aux scripts. Initialement, [la valeur](#) d' un [gestionnaire d'événements](#) doit être définie sur null.
- un **écouteur** , qui est soit null , soit un [écouteur d' événements](#) responsable de l' exécution [de l' algorithme de traitement du gestionnaire d' événements](#) . Initialement, [l'écouteur](#) d' un [gestionnaire d'événements](#) doit être défini sur null.

Les gestionnaires d'événements sont exposés de deux manières.

La première méthode, commune à tous les gestionnaires d'événements, consiste en un [attribut IDL de gestionnaire d'événements](#) .

La deuxième méthode consiste à utiliser un [attribut de contenu de gestionnaire d'événements](#) . Les gestionnaires d'événements sur [les éléments HTML](#) et certains des gestionnaires d'événements sur [Window](#) les objets sont exposés de cette manière.

Pour ces deux manières, le [gestionnaire d'événements](#) est exposé via un **name** , qui est une chaîne qui commence toujours par " on" et est suivie du nom de l'événement auquel le gestionnaire est destiné.

---

La plupart du temps, l'objet qui expose un [gestionnaire d'événement](#) est le même que l'objet sur lequel l' [écouteur d'événement](#) correspondant est ajouté. Cependant, les éléments [body](#) et [frameset](#) exposent plusieurs [gestionnaires d'événements](#) qui agissent sur l' [Window](#) objet de l'élément, s'il en existe un. Dans les deux cas, nous appelons l'objet un [gestionnaire d'événements](#) qui agit sur la **cible** de ce [gestionnaire d'événements](#) .

Pour **déterminer la cible d'un gestionnaire d'événements** , étant donné un [EventTarget](#) objet *eventTarget* sur lequel le [gestionnaire d'événements](#) est exposé et un *nom* [de gestionnaire d'événements name](#) , les étapes suivantes sont suivies :

1. Si *eventTarget* n'est pas un [body](#) élément ou un [frameset](#) élément, alors retourne *eventTarget* .
2. Si *name* n'est pas le nom d'un membre d'attribut de l' [WindowEventHandlers](#) interface mixin et que l' [Window](#) [ensemble de gestionnaires d'événements de l'élément body -reflecting](#) ne [contient pas](#) *name* , alors retournez *eventTarget* .
3. Si [le document de nœud](#) de *eventTarget* n'est pas un [document actif](#) , alors renvoyez null.

*Cela peut arriver si cet objet est un [body](#) élément sans objet correspondant [Window](#), par exemple.  
Cette vérification n'empêche pas nécessairement [body](#) et [frameset](#) les éléments qui ne sont pas [l'élément body](#) de leur [document nœud](#) d'atteindre l'étape suivante. En particulier, un [body](#) élément créé dans un [document actif](#) (peut-être avec `document.createElement()`) mais non [connecté](#) aura également son [Window](#) objet correspondant comme [cible](#) de plusieurs [gestionnaires d'événements](#) exposés à travers lui.*

4. Renvoie [l'objet global pertinent](#) du [document de nœud](#) de *eventTarget* .

---

Chaque [EventTarget](#) objet pour lequel un ou plusieurs [gestionnaires d'événements](#) sont spécifiés est associé à une **mappe de gestionnaires d'événements**, qui est une [mappe](#) de chaînes représentant [des noms](#) de [gestionnaires d'événements](#) vers [des gestionnaires d'événements](#).

Lorsqu'un [EventTarget](#) objet qui a un ou plusieurs [gestionnaires d'événements](#) spécifiés est créé, sa [carte de gestionnaires d'événements](#) doit être initialisée de sorte qu'elle contienne une [entrée](#) pour chaque [gestionnaire d'événements](#) qui a cet objet comme [cible](#), avec [des éléments](#) dans ces [gestionnaires d'événements](#) définis sur leurs valeurs initiales.

*L'ordre des [entrées](#) de [la carte du gestionnaire d'événements](#) peut être arbitraire. Il n'est pas observable à travers les algorithmes qui opèrent sur la carte. [Les entrées](#) ne sont pas créées dans la [carte de gestionnaire d'événements](#) d'un objet pour [les gestionnaires d'événements](#) qui sont simplement exposés sur cet objet, mais qui ont un autre objet comme [cible](#).*

---

Un **attribut IDL de gestionnaire d'événements** est un attribut IDL pour un [gestionnaire d'événements](#) spécifique. Le nom de l'attribut IDL est le même que le [nom](#) du [gestionnaire d'événements](#).

Le getter d'un [attribut IDL de gestionnaire d'événements](#) portant le nom *name*, lorsqu'il est appelé, doit exécuter ces étapes :

1. Soit *eventTarget* le résultat de [la détermination de la cible d'un gestionnaire d'événements](#) en fonction de cet objet et de ce *nom*.
2. Si *eventTarget* est null, renvoie null.
3. Renvoie le résultat de [l'obtention de la valeur actuelle du gestionnaire d'événements](#) avec *eventTarget* et *name*.

Le setter d'un [attribut IDL de gestionnaire d'événements](#) avec le nom *name*, lorsqu'il est appelé, doit exécuter ces étapes :

1. Soit *eventTarget* le résultat de [la détermination de la cible d'un gestionnaire d'événements](#) en fonction de cet objet et de ce *nom*.
2. Si *eventTarget* est null, alors retournez.

3. Si la valeur donnée est null, [désactivez un gestionnaire d'événements](#) donné *eventTarget* et *name* .
4. Sinon:
  1. Soit *handlerMap* la [carte du gestionnaire d'événements](#) de *eventTarget* .
  2. Soit *eventHandler* être *handlerMap* [ *nom* ].
  3. Définissez [la valeur](#) de *eventHandler* sur la valeur donnée.
  4. [Activez un gestionnaire d'événements](#) donné *eventTarget* et *name* .

Certains [attributs IDL de gestionnaire d'événements](#) ont des exigences supplémentaires, en particulier l' [onmessage](#) attribut d' [MessagePort](#) objets.

---

Un **attribut de contenu de gestionnaire d'événements** est un attribut de contenu pour un [gestionnaire d'événements](#) spécifique . Le nom de l'attribut de contenu est le même que le [nom](#) du [gestionnaire d'événements](#) .

[Les attributs de contenu du gestionnaire d'événements](#) , lorsqu'ils sont spécifiés, doivent contenir du code JavaScript valide qui, une fois analysé, correspondrait à la production [FunctionBody](#) après l'insertion automatique du point-virgule .

[Les étapes de changement d'attribut](#) suivantes sont utilisées pour synchroniser entre [les attributs de contenu du gestionnaire d'événements](#) et [les gestionnaires d'événements](#) : [\[DOM\]](#)

1. Si *namespace* n'est pas null, ou *localName* n'est pas le nom d'un [attribut de contenu de gestionnaire d'événements](#) sur *element* , alors retournez.
2. Soit *eventTarget* le résultat de [la détermination de la cible d'un gestionnaire d'événements](#) donné *element* et *localName* .
3. Si *eventTarget* est null, alors retournez.
4. Si *la valeur* est nulle, [désactivez un gestionnaire d'événements](#) donné *eventTarget* et *localName* .
5. Sinon:
  1. Si le [comportement en ligne de l'élément doit-il être bloqué par la politique de sécurité du contenu ?](#) algorithm renvoie " `Blocked`" lorsqu'il est exécuté sur *element* , " `script attribute`" et *value* , puis renvoie. [\[CSP\]](#)

2. Soit *handlerMap* la [carte du gestionnaire d'événements](#) de *eventTarget* .
3. Soit *eventHandler* être *handlerMap* [ *localName* ] .
4. Soit *location* l'emplacement du script qui a déclenché l'exécution de ces étapes.
5. Définissez [la valeur](#) de *eventHandler* sur la *valeur* / l'emplacement [du gestionnaire brut interne non compilé](#) .
6. [Activez un gestionnaire d'événements](#) donné *eventTarget* et *localName* .

*Selon la norme DOM, ces étapes sont exécutées même si *oldValue* et *value* sont identiques (définissant un attribut sur sa valeur actuelle), mais pas si *oldValue* et *value* sont tous les deux nuls (supprimant un attribut qui n'existe pas actuellement). [\[DOM\]](#)*

---

Pour **désactiver un gestionnaire d'événements** à partir d'un [EventTarget](#) objet *eventTarget* et d'un *nom* de chaîne correspondant au [nom](#) d'un [gestionnaire d'événements](#) , exécutez ces étapes :

1. Soit *handlerMap* la [carte du gestionnaire d'événements](#) de *eventTarget* .
2. Soit *eventHandler* être *handlerMap* [ *nom* ] .
3. Définissez la [valeur](#) de *eventHandler* sur null.
4. Soit *listener* le [listener](#) d' *eventHandler* .
5. Si *listener* n'est pas null, [supprimez un écouteur d'événement](#) avec *eventTarget* et *listener* .
6. Définissez l' [écouteur](#) de *eventHandler* sur null.

Pour **effacer tous les écouteurs et gestionnaires d'événements** donnés à un [EventTarget](#) objet *eventTarget* , exécutez ces étapes :

1. Si *eventTarget* a une [mappe de gestionnaire d'événement](#) associée , alors pour chaque *nom* → *eventHandler* de [la mappe de gestionnaire d'événement](#) associée à *eventTarget* , [désactivez un gestionnaire d'événement](#) donné *eventTarget* et *name* .
2. [Supprimez tous les écouteurs d'événements](#) donnés *eventTarget* .

Cet algorithme est utilisé pour définir `document.open()`.

Pour **activer un gestionnaire d'événements** à partir d'un `EventTarget` objet `eventTarget` et d'un `nom` de chaîne correspondant au `nom` d'un `gestionnaire d'événements`, exécutez ces étapes :

1. Soit `handlerMap` la `carte du gestionnaire d'événements` de `eventTarget`.
2. Soit `eventHandler` être `handlerMap [ nom ]`.
3. Si `l'écouteur` de `eventHandler` n'est pas nul, alors retournez.
4. Soit `callback` le résultat de la création d'une `EventListener` instance Web IDL représentant une référence à une fonction d'un argument qui exécute les étapes de `l'algorithme de traitement du gestionnaire d'événements`, étant donné `eventTarget`, `name` et son argument.

Le `contexte de rappel` `EventListener` de peut être arbitraire ; il n'a pas d'incidence sur les étapes de `l'algorithme de traitement du gestionnaire d'événements`. `[DOM]`

*Le rappel n'est absolument pas le `gestionnaire d'événements` lui-même. Chaque `gestionnaire d'événements` finit par enregistrer le même rappel, l'algorithme défini ci-dessous, qui se charge d'invoquer le bon code et de traiter la valeur de retour du code.*

5. Soit `listener` un nouvel `écouteur d'événements` dont `le type` est le **type d'événement du gestionnaire d'événements** correspondant à `eventHandler` et `callback` est `callback`.

*Pour être clair, un `écouteur d'événement` est différent d'un `EventListener`.*

6. `Ajoutez un écouteur d'événement` avec `eventTarget` et `listener`.
7. Définissez `l'écouteur` de `eventHandler` sur `listener`.

*L'enregistrement de l'écouteur d'événements se produit uniquement si la `valeur` du `gestionnaire d'événements` est définie sur une valeur non nulle et que le `gestionnaire d'événements` n'est pas déjà activé. Étant donné que les écouteurs sont appelés dans l'ordre dans lequel ils ont été enregistrés, en supposant qu'aucune `désactivation` n'a eu lieu, l'ordre des écouteurs d'événement pour un type d'événement particulier sera toujours :*

1. les écouteurs d'événements enregistrés avec `addEventListener()` avant la première fois que la `valeur` du `gestionnaire d'événements` a été définie sur non nulle
2. puis le rappel auquel il est actuellement défini, le cas échéant

**3. et enfin les écouteurs d'événements enregistrés avec `addEventListener()` après la première fois que la valeur du gestionnaire d'événements a été définie sur non nulle.**

Cet exemple illustre l'ordre dans lequel les écouteurs d'événement sont appelés. Si le bouton de cet exemple est cliqué par l'utilisateur, la page affichera quatre alertes, avec le texte "UNE", "DEUX", "TROIS" et "QUATRE" respectivement.

```
<button id="test">Start Demo</button>

<script>
  var button = document.getElementById('test');

  button.addEventListener('click', function () { alert('ONE') }, false);

  button.setAttribute('onclick', "alert('NOT CALLED')"); // event handler listener is registered here

  button.addEventListener('click', function () { alert('THREE') }, false);

  button.onclick = function () { alert('TWO'); };

  button.addEventListener('click', function () { alert('FOUR') }, false);
</script>
```

Cependant, dans l'exemple suivant, le gestionnaire d'événements est désactivé après son activation initiale (et son écouteur d'événements est supprimé), avant d'être réactivé ultérieurement. La page affichera cinq alertes avec "UNE", "DEUX", "TROIS", "QUATRE" et "CINQ" respectivement, dans l'ordre.

```
<button id="test">Start Demo</button>

<script>
  var button = document.getElementById('test');

  button.addEventListener('click', function () { alert('ONE') }, false);

  button.setAttribute('onclick', "alert('NOT CALLED')"); // event handler is activated here

  button.addEventListener('click', function () { alert('TWO') }, false);

  button.onclick = null; // but deactivated here

  button.addEventListener('click', function () { alert('THREE') }, false);

  button.onclick = function () { alert('FOUR'); }; // and re-activated here

  button.addEventListener('click', function () { alert('FIVE') }, false);
</script>
```

**Les interfaces implémentées par l'objet événement n'influencent pas le déclenchement ou non d'un gestionnaire d'événements.**



**L'algorithme de traitement du gestionnaire d'événements** pour un `EventTarget` objet `eventTarget`, un *nom* de chaîne représentant le `nom` d'un [gestionnaire d'événements](#) et un événement `Event` d'objet est le suivant :

1. Soit *callback* le résultat de [l'obtention de la valeur actuelle du gestionnaire d'événements](#) donné `eventTarget` et *name* .
2. Si *le rappel* est nul, alors retour.
3. Laissez *la gestion spéciale des événements d'erreur* être vraie si l'événement est un `ErrorEvent` objet, l'événement est et l'événement implémente le mixin. Sinon, laissez *la gestion des événements d'erreur spéciale* être fausse. `typeerrorcurrentTargetWindowOrWorkerGlobalScope`
4. Traitez l' événement `Event` d'objet comme suit :

**Si la gestion des événements d'erreur spéciale est vraie**

[Appeler](#) *le rappel* avec cinq arguments, le premier ayant la valeur de l' attribut de l' événement , le second ayant la valeur de l'attribut de l' événement , le troisième ayant la valeur de l' attribut de l' événement , le quatrième ayant la valeur de l' attribut de l' événement , le cinquième ayant la valeur de l'attribut `event's` , et avec le [rappel cette valeur](#) est définie sur `event's` . Soit la *valeur de retour* la valeur de retour du *rappel*. [\[WEBIDL\]](#) `messagefilenamelinenocolnoerrorcurrentTarget`

**Sinon**

[Appelez](#) *le rappel* avec un argument, dont la valeur est l' `Event` objet `event` , avec le [rappel cette valeur](#) définie sur `event's` `currentTarget`. Soit la *valeur de retour* la valeur de retour du *rappel*. [\[WEBIDL\]](#)

Si une exception est levée par le *rappel*, terminez ces étapes et laissez l'exception se propager. (Il se propagera à la [logique de répartition des événements DOM](#) , qui signalera ensuite [l'exception](#) .)

5. Traiter *la valeur de retour* comme suit :

**Si `event` est un `BeforeUnloadEvent` objet et `event's type` est `beforeunload`**

Dans ce cas, le type [de l'attribut IDL du gestionnaire d'événements](#) sera `OnBeforeUnloadEventHandler`, donc la valeur de retour aura été convertie en `null` ou en a `DOMString`.

Si la *valeur de retour* n'est pas nulle, alors :

1. Définir [l'indicateur d'annulation](#) de l'événement .
2. Si la valeur de l'attribut de l' événement `returnValue` est la chaîne vide, définissez la valeur de l'attribut de l'événement `returnValue` sur la *valeur de retour* .

**Si la gestion des événements d'erreur spéciale est vraie**



Si la valeur de retour est true, définissez [l'indicateur d'annulation](#) de l'événement .

### Sinon

Si la valeur de retour est false, alors définissez [l'indicateur d'annulation](#) de l'événement .

*Si nous sommes arrivés à cette clause "Otherwise" parce que l'événement est mais que l'événement n'est pas un objet, alors la valeur de retour ne sera jamais fausse, car dans de tels cas, [type](#) la valeur de retour aura été forcée en null ou en [a .beforeunloadBeforeUnloadEventDOMString](#)*

---

Le [EventHandler](#) type de fonction de rappel représente un rappel utilisé pour les gestionnaires d'événements. Il est représenté dans Web IDL comme suit :

```
[LegacyTreatNonObjectAsNull]
```

```
callback EventHandlerNonNull = any (Event event);
```

```
typedef EventHandlerNonNull? EventHandler;
```

*En JavaScript, tout [Function](#) objet implémente cette interface.*

Par exemple, le fragment de document suivant :

```
<body onload="alert(this)" onclick="alert(this)">
```

...conduit à une alerte disant " [object Window]" lorsque le document est chargé, et une alerte disant " [object HTMLBodyElement]" chaque fois que l'utilisateur clique sur quelque chose dans la page.

*La valeur de retour de la fonction détermine si l'événement est annulé ou non : comme décrit ci-dessus, si la valeur de retour est fausse, l'événement est annulé.*

*Il existe deux exceptions dans la plateforme, pour des raisons historiques :*

- Les [onerror](#) gestionnaires sur les objets globaux, où retourner true annule l'événement.
- Le [onbeforeunload](#) gestionnaire, où le retour de toute valeur non nulle et non indéfinie annulera l'événement.

Pour des raisons historiques, le [onerror](#) gestionnaire a différents arguments :

```
[LegacyTreatNonObjectAsNull]
```

```
callback OnErrorEventHandlerNonNull = any ((Event or DOMString)  
event, optional DOMString source, optional unsigned long lineno,  
optional unsigned long colno, optional any error);
```

```
typedef OnErrorEventHandlerNonNull? OnErrorEventHandler;
```

```
window.onerror = (message, source, lineno, colno, error) => { ... };
```

De même, le [onbeforeunload](#) gestionnaire a une valeur de retour différente :

```
[LegacyTreatNonObjectAsNull]
```

```
callback OnBeforeUnloadEventHandlerNonNull = DOMString? (Event  
event);
```

```
typedef OnBeforeUnloadEventHandlerNonNull?
```

```
OnBeforeUnloadEventHandler;
```

---

Un **gestionnaire interne brut non compilé** est un tuple contenant les informations suivantes :

- Un corps de script non compilé
- Un emplacement d'où provient le corps du script, au cas où une erreur doit être signalée

Lorsque l'agent utilisateur doit **obtenir la valeur actuelle du gestionnaire d'événements** à partir d'un [EventTarget](#) objet *eventTarget* et d'un *nom* de chaîne correspondant au [nom](#) d'un [gestionnaire d'événements](#) , il doit exécuter ces étapes :

1. Soit *handlerMap* la [carte du gestionnaire d'événements](#) de *eventTarget* .
2. Soit *eventHandler* être *handlerMap* [ *nom* ].
3. Si [la valeur](#) de *eventHandler* est un [gestionnaire brut interne non compilé](#) , alors :
  1. Si *eventTarget* est un élément, alors laissez *element* être *eventTarget* , et *document* être [le nœud](#)

[document](#) de l'élément . Sinon, *eventTarget* est un objet, laissez *element* être null et *document* être [associé](#) à *eventTarget* .[WindowDocument](#)

2. Si [le script est désactivé](#) pour *document* , renvoie null.
3. Soit *body* le corps du script non compilé dans la [valeur](#) de *eventHandler* .
4. Soit *location* l'emplacement d'origine du corps du script, tel qu'indiqué par [la valeur](#) de *eventHandler* .
5. Si *element* n'est pas nul et que *element* a un [propriétaire de formulaire](#) , laissez le *propriétaire du formulaire* être ce [propriétaire du formulaire](#) . Sinon, laissez le *propriétaire du formulaire* être nul.
6. Soit *settings object* l' [objet de paramètres pertinent](#) du *document* .
7. Si *body* n'est pas analysable en tant que [FunctionBody](#) ou si l'analyse détecte une [erreur précoce](#) , suivez ces sous-étapes :
  1. Définissez la [valeur](#) de *eventHandler* sur null.

*Cela ne désactive pas le gestionnaire d'événements, ce qui supprime également l'écouteur du gestionnaire d'événements (s'il est présent).*

2. [Signalez l'erreur](#) pour le [script](#) approprié et avec la position appropriée (numéro de ligne et numéro de colonne) donnée par *location* , en utilisant l' [objet global](#) de l'*objet settings* . Si l'erreur n'est toujours [pas traitée](#) après cela, l'erreur peut être signalée à une console de développeur.
  3. Renvoie nul.
8. Poussez [le contexte d'exécution du domaine](#) de l'*objet de paramètres* sur la [pile de contexte d'exécution JavaScript](#) ; c'est maintenant le [contexte d'exécution de JavaScript en cours d'exécution](#) .

*Cela est nécessaire pour que l'appel ultérieur de [OrdinaryFunctionCreate](#) ait lieu dans le [domaine](#) correct .*

9. Soit *function* le résultat de l'appel [de OrdinaryFunctionCreate](#) , avec les arguments :

***functionPrototype***

[%Function.prototype%](#)

***texte source***

**Si *name* est [onerror](#) et *eventTarget* est un [Window](#) objet**

La chaîne formée en concaténant " *function* ", *name* , " (event, source, lineno, colno, error) {", U+000A LF, *body* , U+000A LF et " }".

## Sinon

La chaîne formée en concaténant " `function` ", *name* , " (event) {", U+000A LF, *body* , U+000A LF et " }".

### Liste de paramètres

Si *name* est onerror et *eventTarget* est un Window objet

Laissez la fonction avoir cinq arguments, nommés `event`, , ,  
et `.sourcelinenocolnoerror`

## Sinon

Laissez la fonction avoir un seul argument appelé `event`.

### corps

Le résultat de l'analyse *du corps* ci-dessus.

### ceMode

ceci non lexical

### portée

1. Soit *domaine* le domaine de l'objet paramètres .
2. Soit *scope* le *domaine* .[[GlobalEnv]].
3. Si *eventHandler* est le gestionnaire d'événements d'un élément ,  
définissez *scope* sur NewObjectEnvironment ( *document* , true, *scope* ).  
  
(Sinon, *eventHandler* est le gestionnaire d'événementsWindow d'un  
objet .)
4. Si le propriétaire du formulaire n'est pas nul, définissez la  
*portée* sur NewObjectEnvironment ( *propriétaire du formulaire* ,  
true, *scope* ).
5. Si *element* n'est pas null,  
définissez *scope* sur NewObjectEnvironment ( *element* , true, *scope* ).
6. *Portée* de retour .
10. Supprimez le contexte d'exécution du domaine de l'objet de  
*paramètres* de la pile de contexte d'exécution JavaScript .
11. Définissez la fonction .[[ScriptOrModule]] sur null.

Ceci est fait parce que le comportement par défaut, consistant à associer la fonction créée au script le plus proche sur la pile, peut conduire à des résultats dépendants du chemin. Par exemple, un gestionnaire d'événements qui est d'abord invoqué par l'interaction de l'utilisateur se retrouverait avec null [[ScriptOrModule]] (depuis lors, cet algorithme serait d'abord invoqué lorsque le script actif est nul), alors que celui qui est d'abord invoqué en envoyant un événement du script aurait son [[ScriptOrModule]] défini sur ce script.

Au lieu de cela, nous définissons toujours `[[ScriptOrModule]]` sur `null`. C'est plus intuitif de toute façon; l'idée que le premier script qui distribue un événement est en quelque sorte responsable du code du gestionnaire d'événements est douteuse.

En pratique, cela n'affecte que la résolution des URL relatives via `import()`, qui consultent l'[URL de base](#) du script associé. La suppression de `[[ScriptOrModule]]` signifie que [HostLoadImportedModule](#) reviendra à l'[URL de base de l'API](#) de l'[objet de paramètres actuel](#).

12. Définissez [la valeur](#) de `eventHandler` sur le résultat de la création d'un objet de fonction de rappel Web IDL dont la référence d'objet est la fonction et dont [le contexte de rappel](#) est l'objet de paramètres `.EventHandler`

4. Renvoie [la valeur](#) de `eventHandler`.

#### 8.1.8.2 Gestionnaires d'événements sur des éléments, [Document](#) des objets et [Window](#) des objets

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants) qui doivent être pris en charge par tous [les éléments HTML](#), en tant [qu'attributs de contenu de gestionnaire d'événements](#) et [attributs IDL de gestionnaire d'événements](#); et qui doivent être pris en charge par tous les objets [Document](#) et [Window](#), en tant [qu'attributs IDL du gestionnaire d'événements](#):

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<code>onabort</code> ✓ <a href="#">MDN</a>	<a href="#">abort</a>
<code>onauxclick</code> <a href="#">MDN</a>	<a href="#">auxclick</a>
<code>onbeforeinput</code>	<a href="#">beforeinput</a>
<code>onbeforematch</code>	<a href="#">beforematch</a>
<code>onbeforetoggle</code>	<a href="#">beforetoggle</a>
<code>oncancel</code> ✓ <a href="#">MDN</a>	<a href="#">cancel</a>
<code>oncanplay</code> ✓ <a href="#">MDN</a>	<a href="#">canplay</a>
<code>oncanplaythrough</code> ✓ <a href="#">MDN</a>	<a href="#">canplaythrough</a>
<code>onchange</code> ✓ <a href="#">MDN</a>	<a href="#">change</a>
<code>onclick</code>	<a href="#">click</a>

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
✓ MDN	
onclose	<a href="#">close</a>
oncontextlost	<a href="#">contextlost</a>
oncontextmenu	<a href="#">contextmenu</a>
oncontextrestored	<a href="#">contextrestored</a>
oncopy	<a href="#">copy</a>
✓ MDN	
oncuechange	<a href="#">cuechange</a>
✓ MDN	
oncut	<a href="#">cut</a>
✓ MDN	
ondblclick	<a href="#">dblclick</a>
✓ MDN	
ondrag	<a href="#">drag</a>
ondragend	<a href="#">dragend</a>
ondragenter	<a href="#">dragenter</a>
ondragleave	<a href="#">dragleave</a>
ondragover	<a href="#">dragover</a>
ondragstart	<a href="#">dragstart</a>
ondrop	<a href="#">drop</a>
ondurationchange	<a href="#">durationchange</a>
✓ MDN	
onemptied	<a href="#">emptied</a>
✓ MDN	
onended	<a href="#">ended</a>
✓ MDN	
onformdata	<a href="#">formdata</a>
oninput	<a href="#">input</a>
✓ MDN	
oninvalid	<a href="#">invalid</a>
onkeydown	<a href="#">keydown</a>
✓ MDN	
onkeypress	<a href="#">keypress</a>
onkeyup	<a href="#">keyup</a>
✓ MDN	
onloadeddata	<a href="#">loadeddata</a>
✓ MDN	
onloadedmetadata	<a href="#">loadedmetadata</a>
✓ MDN	
onloadstart	<a href="#">loadstart</a>
✓ MDN	

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
onmousedown ✓ MDN	<u>mousedown</u>
onmouseenter ✓ MDN	<u>mouseenter</u>
onmouseleave ✓ MDN	<u>mouseleave</u>
onmousemove ✓ MDN	<u>mousemove</u>
onmouseout ✓ MDN	<u>mouseout</u>
onmouseover ✓ MDN	<u>mouseover</u>
onmouseup ✓ MDN	<u>mouseup</u>
onpaste ✓ MDN	<u>paste</u>
onpause ✓ MDN	<u>pause</u>
onplay ✓ MDN	<u>play</u>
onplaying ✓ MDN	<u>playing</u>
onprogress ✓ MDN	<u>progress</u>
onratechange ✓ MDN	<u>ratechange</u>
onreset ✓ MDN	<u>reset</u>
onsecuritypolicyviolation ✓ MDN	<u>securitypolicyviolation</u>
onseeked ✓ MDN	<u>seeked</u>
onseeking ✓ MDN	<u>seeking</u>
onselect ✓ MDN	<u>select</u>
onslotchange ✓ MDN	<u>slotchange</u>
onstalled ✓ MDN	<u>stalled</u>
onsubmit ✓ MDN	<u>submit</u>
onsuspend ✓ MDN	<u>suspend</u>

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
✓ MDN	
on <del>time</del> update	<u>timeupdate</u>
✓ MDN	
on <del>toggle</del>	<u>toggle</u>
on <del>volume</del> change	<u>volumechange</u>
✓ MDN	
on <del>waiting</del>	<u>waiting</u>
✓ MDN	
onwebkitanimationend	webkitAnimationEnd
onwebkitanimationiteration	webkitAnimationIteration
onwebkitanimationstart	webkitAnimationStart
onwebkittransitionend	webkitTransitionEnd
on <del>wheel</del>	<u>wheel</u>
✓ MDN	

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge par tous [les éléments HTML](#) autres que les éléments `body` et `frameset` , en tant [qu'attributs de contenu de gestionnaire d'événements](#) et [attributs IDL de gestionnaire d'événements](#) ; qui doivent être pris en charge par tous [Document](#) les objets, en tant [qu'attributs IDL du gestionnaire d'événements](#) ; et qui doivent être pris en charge par tous [Window](#) les objets, en tant [qu'attributs IDL de gestionnaire d'événements](#) sur les [Window](#) objets eux-mêmes, et avec [les attributs de contenu de gestionnaire d'événements](#) correspondants et [les attributs IDL de gestionnaire d'événements](#) exposés sur tous `body` et `frameset` les éléments qui appartiennent à cet [Window](#) objet [associé Document](#) :

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
on <del>blur</del>	<u>blur</u>
✓ MDN	
on <del>error</del>	<u>error</u>
✓ MDN	
on <del>focus</del>	<u>focus</u>
✓ MDN	
on <del>load</del>	<u>load</u>
on <del>resize</del>	<u>resize</u>
on <del>scroll</del>	<u>scroll</u>
✓ MDN	



Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
onscrollend	scrollend

Nous appelons l' ensemble des noms des gestionnaires d'événements répertoriés dans la première colonne de ce tableau l' **Window**ensemble de gestionnaires d'événements de l'élément de corps -reflecting .

Voici les gestionnaires d'événements (et leurs types d'événements de gestionnaire d'événements correspondants ) qui doivent être pris en charge par Windowles objets, en tant qu'attributs IDL de gestionnaire d'événements sur les Windowobjets eux-mêmes, et avec les attributs de contenu de gestionnaire d'événements correspondants et les attributs IDL de gestionnaire d'événements exposés sur tous body les framesetéléments qui appartiennent à l' Windowobjet associéDocument :

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
onafterprint ✓ MDN	afterprint
onbeforeprint ✓ MDN	beforeprint
onbeforeunload ✓ MDN	beforeunload
onhashchange ✓ MDN	hashchange
onlanguagechange ✓ MDN	languagechange
onmessage ✓ MDN	message
onmessageerror ✓ MDN	messageerror
onoffline ✓ MDN	offline
ononline ✓ MDN	online
onpagehide	pagehide
onpageshow	pageshow
onpopstate ✓ MDN	popstate
onrejectionhandled	rejectionhandled

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
✓ MDN	
onstorage	storage
✓ MDN	
onunhandledrejection	unhandledrejection
✓ MDN	
onunload	unload
✓ MDN	

Cette liste de [gestionnaires d'événements](#) est réifiée en tant [qu'attributs IDL de gestionnaire d'événements](#) via l' [WindowEventHandlers](#) interface mixin.

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge sur [Document](#) les objets en tant [qu'attributs IDL de gestionnaire d'événements](#) :

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
onreadystatechange	readystatechange
onvisibilitychange	visibilitychange
✓ MDN	

#### 8.1.8.2.1 Définitions IDL

```
interface mixin GlobalEventHandlers {
```

```
  attribute EventHandler onabort;
```

```
  attribute EventHandler onauxclick;
```

```
  attribute EventHandler onbeforeinput;
```

```
  attribute EventHandler onbeforematch;
```

```
  attribute EventHandler onblur;
```

```
  attribute EventHandler oncancel;
```

```
attribute EventHandler oncanplay;
```

```
attribute EventHandler oncanplaythrough;
```

```
attribute EventHandler onchange;
```

```
attribute EventHandler onclick;
```

```
attribute EventHandler onclose;
```

```
attribute EventHandler oncontextlost;
```

```
attribute EventHandler oncontextmenu;
```

```
attribute EventHandler oncontextrestored;
```

```
attribute EventHandler oncopy;
```

```
attribute EventHandler oncuechange;
```

```
attribute EventHandler oncut;
```

```
attribute EventHandler ondblclick;
```

```
attribute EventHandler ondrag;
```

```
attribute EventHandler ondragend;
```

```
attribute EventHandler ondragenter;
```

```
attribute EventHandler ondragleave;
```

```
attribute EventHandler ondragover;
```

```
attribute EventHandler ondragstart;
```

```
attribute EventHandler ondrop;
```

```
attribute EventHandler ondurationchange;
```

```
attribute EventHandler onemptied;
```

```
attribute EventHandler onended;
```

attribute [OnErrorEventHandler](#) [onerror](#);

attribute [EventHandler](#) [onfocus](#);

attribute [EventHandler](#) [onformdata](#);

attribute [EventHandler](#) [oninput](#);

attribute [EventHandler](#) [oninvalid](#);

attribute [EventHandler](#) [onkeydown](#);

attribute [EventHandler](#) [onkeypress](#);

attribute [EventHandler](#) [onkeyup](#);

attribute [EventHandler](#) [onload](#);

attribute [EventHandler](#) [onloadeddata](#);

attribute [EventHandler](#) [onloadedmetadata](#);

attribute [EventHandler](#) [onloadstart](#);

attribute [EventHandler](#) [onmousedown](#);

[[LegacyLenientThis](#)] attribute [EventHandler](#) [onmouseenter](#);

[[LegacyLenientThis](#)] attribute [EventHandler](#) [onmouseleave](#);

attribute [EventHandler](#) [onmousemove](#);

attribute [EventHandler](#) [onmouseout](#);

attribute [EventHandler](#) [onmouseover](#);

attribute [EventHandler](#) [onmouseup](#);

attribute [EventHandler](#) [onpaste](#);

attribute [EventHandler](#) [onpause](#);

attribute [EventHandler](#) [onplay](#);

attribute [EventHandler](#) [onplaying](#);

attribute [EventHandler](#) [onprogress](#);

attribute [EventHandler](#) [onratechange](#);

attribute [EventHandler](#) [onreset](#);

attribute [EventHandler](#) [onresize](#);

attribute [EventHandler](#) [onscroll](#);

attribute [EventHandler](#) [onscrollend](#);

attribute [EventHandler](#) [onsecuritypolicyviolation](#);

attribute [EventHandler](#) [onseeked](#);

attribute [EventHandler](#) [onseeking](#);

attribute [EventHandler](#) [onselect](#);

attribute [EventHandler](#) [onslotchange](#);

attribute [EventHandler](#) [onstalled](#);

attribute [EventHandler](#) [onsubmit](#);

attribute [EventHandler](#) [onsuspend](#);

attribute [EventHandler](#) [ontimeupdate](#);

attribute [EventHandler](#) [ontoggle](#);

attribute [EventHandler](#) [onvolumechange](#);

attribute [EventHandler](#) [onwaiting](#);

attribute [EventHandler](#) [onwebkitanimationend](#);

attribute [EventHandler](#) [onwebkitanimationiteration](#);

attribute [EventHandler](#) [onwebkitanimationstart](#);

```
attribute EventHandler onwebkittransitionend;
```

```
attribute EventHandler onwheel;
```

```
};
```

```
interface mixin WindowEventHandlers {
```

```
attribute EventHandler onafterprint;
```

```
attribute EventHandler onbeforeprint;
```

```
attribute OnBeforeUnloadEventHandler onbeforeunload;
```

```
attribute EventHandler onhashchange;
```

```
attribute EventHandler onlanguagechange;
```

```
attribute EventHandler onmessage;
```

```
attribute EventHandler onmessageerror;
```

```
attribute EventHandler onoffline;
```

```
attribute EventHandler ononline;
```

```
attribute EventHandler onpagehide;
```

```
attribute EventHandler onpageshow;
```

```
attribute EventHandler onpopstate;
```

```
attribute EventHandler onrejectionhandled;
```

```
attribute EventHandler onstorage;
```

```
attribute EventHandler onunhandledrejection;
```

```
attribute EventHandler onunload;
```

```
};
```

### 8.1.8.3 Déclenchement d'événement

Certaines opérations et méthodes sont définies comme des événements de déclenchement sur des éléments. Par exemple, la `click()` méthode sur l'`HTMLElement` interface est définie comme déclenchant un `click` événement sur l'élément. [\[UIÉVÉNEMENTS\]](#)

**Le déclenchement d'un événement de pointeur synthétique nommé `e` sur `target` , avec un *indicateur facultatif non approuvé* , signifie exécuter ces étapes :**

1. Soit *événement* le résultat de [la création d'un événement](#) à l'aide de `PointerEvent`.
2. Initialise l'attribut de l'événement `type` à `e` .
3. Initialise les événements et `bubbles` les `cancelable` attributs à `true`.
4. Définir [l'indicateur composé](#) de l'événement .
5. Si l' *indicateur non approuvé* est défini, initialisez l'attribut de l'événement `isTrusted` à `faux`.
6. Initialise les attributs `ctrlKey`, `shiftKey`, `altKey` et `metaKey` de l'événement en fonction de l'état actuel du périphérique d'entrée de clé, le cas échéant (`false` pour toutes les clés qui ne sont pas disponibles).
7. Initialise l'attribut de l'événement `view` à l'objet du [document du noeud](#) de la *cible* , le cas échéant, et `null` sinon. `Window`
8. La méthode de l' événement `getModifierState()` consiste à renvoyer des valeurs décrivant de manière appropriée l'état actuel du périphérique d'entrée à clé.
9. Renvoie le résultat de [l'envoi](#) de l'événement à la *cible* .

**Le déclenchement d'un `click` événement sur la *cible* signifie [le déclenchement d'un événement de pointeur synthétique nommé `click` sur la cible](#) .**

## 8.2 Le `WindowOrWorkerGlobalScope` **mixage**

Le `WindowOrWorkerGlobalScope` mixin est destiné à l'utilisation d'API qui doivent être exposées sur `Window` des `WorkerGlobalScope` objets.

*D'autres normes sont encouragées à l'étendre davantage en utilisant une référence appropriée.* `partial interface mixin WindowOrWorkerGlobalScope { ... };`

```
typedef (DOMString or Function) TimerHandler;
```

```
interface mixin WindowOrWorkerGlobalScope {
```

```
  [Replaceable] readonly attribute USVString origin;
```

```
  readonly attribute boolean isSecureContext;
```

```
  readonly attribute boolean crossOriginIsolated;
```

```
  undefined reportError(any e);
```

```
  // base64 utility methods
```

```
  DOMString btoa(DOMString data);
```

```
  ByteString atob(DOMString data);
```

```
  // timers
```

```
  long setTimeout(TimerHandler handler, optional long timeout =  
0, any... arguments);
```

```
  undefined clearTimeout(optional long id = 0);
```

```
  long setInterval(TimerHandler handler, optional long timeout =  
0, any... arguments);
```

```
  undefined clearInterval(optional long id = 0);
```

```
  // microtask queuing
```

```
  undefined queueMicrotask(VoidFunction callback);
```

```
  // ImageBitmap
```



```
Promise<ImageBitmap> createImageBitmap(ImageBitmapSource image,  
optional ImageBitmapOptions options = {});
```

```
Promise<ImageBitmap> createImageBitmap(ImageBitmapSource image,  
long sx, long sy, long sw, long sh, optional ImageBitmapOptions  
options = {});
```

```
// structured cloning
```

```
any structuredClone(any value, optional  
StructuredSerializeOptions options = {});
```

```
};
```

```
Window includes WindowOrWorkerGlobalScope;
```

```
WorkerGlobalScope includes WindowOrWorkerGlobalScope;
```

**self.isSecureContext**

✓

Retourne si cet objet global représente ou non un [contexte sécurisé](#) . [\[CONTEXTES SÉCURISÉS\]](#)

**self.origin**

✓

Renvoie l' [origine](#) de l'objet global , sérialisé sous forme de chaîne.

**self.crossOriginIsolated**

✓

Renvoie si les scripts s'exécutant dans ce global sont autorisés à utiliser des API qui nécessitent une isolation cross-origine. Cela dépend des en-têtes de réponse HTTP ` [Cross-Origin-Opener-Policy](#) ` et ` [Cross-Origin-Embedder-Policy](#) ` et de la [cross-origin-isolated](#) fonctionnalité " " .

Les développeurs sont fortement encouragés à utiliser `self.origin` over `location.origin`. Le premier renvoie l' [origine](#) de l'environnement, le second l'URL de l'environnement. Imaginez le script suivant s'exécutant dans un document sur `https://stargate.example/`:

```
var frame = document.createElement("iframe")  
frame.onload = function() {  
  var frameWin = frame.contentWindow  
  console.log(frameWin.location.origin) // "null"
```

```
console.log(frameWin.origin) // "https://stargate.example"
}
document.body.appendChild(frame)
```

`self.origin` est un indicateur de sécurité plus fiable.

Les `isSecureContext` étapes du getter consistent à renvoyer `true` si [cet](#) objet de [paramètres pertinent](#) est un [contexte sécurisé](#), ou `false` sinon.

Les `origin` étapes du getter consistent à renvoyer l' [origine](#) de [cet objet de paramètres pertinent](#), [sérialisé](#).

Les `crossOriginIsolated` étapes du getter consistent à renvoyer [la capacité d'isolement cross-origin](#) de [cet](#) objet [de paramètres pertinent](#).

## 8.3 Méthodes utilitaires Base64

Les méthodes [atob\(\)](#) et [btoa\(\)](#) permettent aux développeurs de transformer le contenu vers et depuis l'encodage base64.

*Dans ces API, à des fins mnémotechniques, le "b" peut être considéré comme signifiant "binaire", et le "a" comme "ASCII". En pratique, cependant, pour des raisons principalement historiques, l'entrée et la sortie de ces fonctions sont des chaînes Unicode.*

```
result = self.btoa(data)
```

✓ 

Prend les données d'entrée, sous la forme d'une chaîne Unicode contenant uniquement des caractères dans la plage U+0000 à U+00FF, chacun représentant un octet binaire avec les valeurs 0x00 à 0xFF respectivement, et les convertit dans sa représentation base64, qu'elle renvoie.

Lève une exception `"InvalidCharacterError"` [DOMException](#) si la chaîne d'entrée contient des caractères hors limites.

```
result = self.atob(data)
```

✓ 

Prend les données d'entrée, sous la forme d'une chaîne Unicode contenant des données binaires codées en base64, les décode et renvoie une chaîne composée de caractères dans la plage U+0000 à U+00FF, chacun représentant un octet binaire avec les valeurs 0x00 à 0xFF respectivement, correspondant à ces données binaires.

Lève un `"InvalidCharacterError"` [DOMException](#) si la chaîne d'entrée n'est pas une donnée base64 valide.

La méthode doit lancer un `" "` si *les données* contiennent un caractère dont le point de code est supérieur à U+00FF. Sinon, l'agent utilisateur doit convertir *les données* en une séquence d'octets dont *le n* ième octet est la représentation à huit bits du *n* ième point de code de *data* , puis doit appliquer [l'encodage pardonnant-base64](#) à cette séquence d'octets et renvoyer le résultat.`btoa(data)` `InvalidCharacterError` `DOMException`

Les étapes de la méthode sont :`atob(data)`

1. Soit *decodedData* le résultat de l'exécution [du décodage pardonnant-base64](#) sur *data* .
2. Si *decodedData* est un échec, lancez un `"InvalidCharacterError"` `DOMException` .
3. Renvoie *decodedData* .

[← 8 API d'application Web](#) — [Table des matières](#) — [8.6 Minuteries](#) →

1.
  1. [8.4 Insertion de balisage dynamique](#)
    1. [8.4.1 Ouverture du flux d'entrée](#)
    2. [8.4.2 Fermeture du flux d'entrée](#)
    3. [8.4.3 `document.write\(\)`](#)
    4. [8.4.4 `document.writeln\(\)`](#)
  2. [8.5 Analyse DOM](#)

## 8.4 Insertion de balisage dynamique

*Les API d'insertion dynamique de balisage dans le document interagissent avec l'analyseur, et leur comportement varie donc selon qu'elles sont utilisées avec [des documents HTML](#) (et l' [analyseur HTML](#) ) ou [des documents XML](#) (et l' [analyseur XML](#) ).*

`Document` les objets ont un **compteur d'insertion de balisage dynamique** , qui est utilisé conjointement avec l' algorithme [de création d'un élément pour le jeton](#) afin d'empêcher [les constructeurs d'éléments personnalisés](#) d'utiliser `document.open()` , `document.close()` et `document.write()` lorsqu'ils sont invoqués par l'analyseur. Initialement, le compteur doit être mis à zéro.

### 8.4.1 Ouverture du flux d'entrée

```
document = document.open()
```

✓

Provoque le [Document](#) remplacement sur place, comme s'il s'agissait d'un nouvel [Document](#) objet, mais en réutilisant l'objet précédent, qui est ensuite renvoyé.

Le résultat [Document](#) est associé à un analyseur HTML, qui peut recevoir des données à analyser à l'aide de [document.write\(\)](#).

La méthode n'a aucun effet si le [Document](#) est toujours en cours d'analyse.

Lance un ["InvalidStateError"](#) [DOMException](#) si le [Document](#) est un [document XML](#).

Lève un ["InvalidStateError"](#) [DOMException](#) si l'analyseur est en train d'exécuter un [constructeur d'élément personnalisé](#).

```
window = document.open(url, name, features)
```

Fonctionne comme la [window.open\(\)](#) méthode.

[Document](#) les objets ont un **analyseur actif a été abandonné** booléen, qui est utilisé pour empêcher les scripts d'invoquer les méthodes [document.open\(\)](#) et [document.write\(\)](#) (directement ou indirectement) après l'abandon de [l'analyseur actif du document](#). C'est d'abord faux.

Les **étapes d'ouverture de document**, étant donné un *document*, sont les suivantes :

1. Si *document* est un [document XML](#), lancez une exception ["InvalidStateError"](#) [DOMException](#).
2. Si [le compteur throw-on-dynamic-markup-insertion](#) du *document* est supérieur à 0, lancez un ["InvalidStateError"](#) [DOMException](#).
3. Soit *entryDocument* l'objet [global d'entrée associé](#) [Document](#).
4. Si [l'origine](#) du *document* n'est pas [la même origine](#) que [l'origine](#) de *entryDocument*, lancez un ["SecurityError"](#) [DOMException](#).
5. Si *document* a un [analyseur actif](#) dont [le niveau d'imbrication de script](#) est supérieur à 0, alors renvoie *document*.

*Cela entraîne essentiellement [document.open\(\)](#) son ignorance lorsqu'il est appelé dans un script en ligne trouvé lors de l'analyse, tout en lui permettant d'avoir un effet lorsqu'il est appelé à partir d'une tâche non analyseur, telle qu'un rappel de minuterie ou un gestionnaire d'événements.*

6. De même, si [le compteur de déchargement](#) de *document* est supérieur à 0, alors retournez *document*.

Cela entraîne essentiellement `document.open()` son ignorance lorsqu'il est appelé à partir d'un gestionnaire d'événements `beforeunload`, `pagehide` ou `unload` pendant le `Document` déchargement de .

7. Si l'analyseur actif de `document` a été abandonné est true, alors retourne `document` .

Cela a notamment pour effet `document.open()` d'être ignoré s'il est appelé après le début d'une navigation , mais uniquement lors de l'analyse initiale. Voir le numéro 4723 pour plus d'informations.

8. Si le nœud navigable du `document` n'est pas nul et que la navigation en cours du nœud navigable du `document` est un ID de navigation , alors arrêtez de charger le nœud navigable du `document` .
9. Pour chaque `nœud` descendant inclusif shadow-inclusive de `document` , effacez tous les écouteurs et gestionnaires d'événements donnés `node` .
10. Si `document` est associé `Document` à l' objet global pertinent de `document` , alors effacez tous les écouteurs et gestionnaires d'événements donnés par l' objet global pertinent de `document` .
11. Remplacez all par null dans `document` , sans déclencher d'événements de mutation.
12. Si le `document` est entièrement actif , alors :
  1. Soit `newURL` une copie de l'URL de `entryDocument` .
  2. Si `entryDocument` n'est pas `document` , définissez le fragment de `newURL` sur null.
  3. Exécutez les étapes de mise à jour de l'URL et de l'historique avec `document` et `newURL` .
13. Set `document` 's is initial`about:blank` to false.
14. Si l'indicateur de chargement iframe en cours du `document` est défini, définissez l'indicateur de chargement iframe muet du `document` .
15. Définissez le `document` sur le mode sans bizarrerie .
16. Créez un nouvel analyseur HTML et associez-le à `document` . Il s'agit d'un **analyseur créé par un script** (ce qui signifie qu'il peut être fermé par les méthodes `document.open()` et `document.close()` , et que le tokenizer attendra un appel explicite à `document.close()` avant d'émettre un jeton de fin de fichier). La confiance d'encodage n'est pas pertinente .
17. Définissez le point d'insertion juste avant la fin du flux d'entrée (qui à ce stade sera vide).

18. [Mettez à jour l'état de préparation actuel](#) du *document* sur " loading".

*Cela provoque [readystatechange](#) le déclenchement d'un événement, mais l'événement est en fait inobservable pour le code de l'auteur, en raison de l'étape précédente qui [a effacé tous les écouteurs et gestionnaires d'événements](#) qui pourraient l'observer.*

19. Document de retour .

*Les [étapes d'ouverture du document](#) n'affectent pas si a [Document](#) est [prêt pour les tâches de post-chargement](#) ou [complètement chargé](#) .*

La méthode doit renvoyer le résultat de l'exécution des [étapes d'ouverture du document](#) avec [this](#) .[open \(unused1, unused2\)](#)

*Les arguments inutilisé1 et inutilisé2 sont ignorés, mais conservés dans l'IDL pour permettre au code qui appelle la fonction avec un ou deux arguments de continuer à fonctionner. Ils sont nécessaires en raison des règles [de l'algorithme de résolution de surcharge](#) Web IDL , qui lèveraient une [TypeError](#) exception pour de tels appels si les arguments n'étaient pas là. [whatwg/webidl numéro 581](#) étudie la modification de l'algorithme pour permettre leur suppression. [\[WEBIDL\]](#)*

La méthode doit exécuter ces étapes :[open\(url, name, features\)](#)

1. Si [ce](#) n'est pas [complètement actif](#) , lancez une exception "[InvalidAccessError](#)". [DOMException](#)
2. Renvoie le résultat de l'exécution des [étapes d'ouverture de la fenêtre](#) avec *url* , *name* et *features* .

#### 8.4.2 Fermeture du flux d'entrée

[document.close\(\)](#)

✓

*Ferme le flux d'entrée qui a été ouvert par la [document.open\(\)](#) méthode.*

*Lance un "[InvalidStateError](#)" [DOMException](#) si le [Document](#) est un [document XML](#) .*

*Lève un "[InvalidStateError](#)" [DOMException](#) si l'analyseur est en train d'exécuter un [constructeur d'élément personnalisé](#) .*

La [close\(\)](#) méthode doit exécuter les étapes suivantes :

1. S'il [s'agit](#) d'un [document XML](#) , lancez un "[InvalidStateError](#)" [DOMException](#) .
2. Si [ce](#) compteur [d'insertion de balisage dynamique](#) est supérieur à zéro, lancez un "[InvalidStateError](#)" [DOMException](#) .

3. S'il n'y a pas [d'analyseur créé par un script](#) associé à [this](#) , alors retournez.
4. Insérez un [caractère "EOF" explicite](#) à la fin du [flux d'entrée](#) de l'analyseur .
5. Si [ce script de blocage d'analyse en attente](#) n'est pas nul, alors retournez .
6. Exécutez le tokenizer, en traitant les jetons résultants au fur et à mesure qu'ils sont émis et en s'arrêtant lorsque le tokenizer atteint le [caractère "EOF" explicite](#) ou [tourne la boucle d'événement](#) .

#### 8.4.3 [document.write\(\)](#)

```
document.write(...text)
```



En général, ajoute la ou les chaînes données au [Document](#) flux d'entrée de .

**Cette méthode a un comportement très idiosyncrasique. Dans certains cas, cette méthode peut affecter l'état de l' [analyseur HTML](#) pendant que l'analyseur est en cours d'exécution, ce qui entraîne un DOM qui ne correspond pas à la source du document (par exemple, si la chaîne écrite est la chaîne " " <plaintext> ou " <!-- "). Dans d'autres cas, l'appel peut d'abord effacer la page en cours, comme s'il [document.open\(\)](#) avait été appelé. Dans encore plus de cas, la méthode est simplement ignorée ou lève une exception. Les agents utilisateurs sont [explicitement autorisés à éviter d'exécuter script les éléments insérés via cette méthode](#) . Et pour aggraver les choses, le comportement exact de cette méthode peut dans certains cas dépendre de la latence du réseau, ce qui peut entraîner des pannes très difficiles à déboguer. Pour toutes ces raisons, l'utilisation de cette méthode est fortement déconseillée.**

Lance un `"_InvalidStateError"` [DOMException](#) lorsqu'il est appelé sur [des documents XML](#) .

Lève un `"_InvalidStateError"` [DOMException](#) si l'analyseur est en train d'exécuter un [constructeur d'élément personnalisé](#) .

[Document](#) les objets ont un **compteur ignore-destructive-writes** , qui est utilisé en conjonction avec le traitement des [script](#) éléments pour empêcher les scripts externes de pouvoir utiliser [document.write\(\)](#) pour détruire le document en appelant implicitement [document.open\(\)](#) . Initialement, le compteur doit être mis à zéro.

Les **étapes d'écriture du document** , étant donné un `document` [Document](#) objet et une *entrée* de chaîne , sont les suivantes :

1. Si `document` est un [document XML](#) , lancez un `"_InvalidStateError"` [DOMException](#) .



2. Si [le compteur throw-on-dynamic-markup-insertion](#) du *document* est supérieur à 0, lancez un `"_".InvalidStateError DOMException`
3. Si l'analyseur actif du *document* [a été abandonné](#) est vrai, alors retournez.
4. Si le [point d'insertion](#) n'est pas défini, alors :
  1. Si [le compteur de déchargement](#) du *document* est supérieur à 0 ou si [le compteur d'écritures ignorées-destructrices du](#) *document* est supérieur à 0, alors retour.
  2. Exécutez les [étapes d'ouverture du document](#) avec *document*.
5. Insérez l'entrée dans le [flux d'entrée](#) juste avant le [point d'insertion](#).
6. Si [le script de blocage d'analyse en attente](#) du *document* est nul, alors faites en sorte que l'[analyseur HTML](#) traite *input*, un point de code à la fois, traitant les jetons résultants au fur et à mesure qu'ils sont émis et s'arrêtant lorsque le tokenizer atteint le point d'insertion ou lorsque le traitement du tokenizer est abandonné par l'étape de construction de l'arbre (cela peut arriver si un jeton de balise de fin est émis par le tokenizer). `script`

*Si la `document.write()` méthode a été appelée à partir d'un script s'exécutant en ligne (c'est-à-dire parce que l'analyseur a analysé un ensemble de `script` balises), il s'agit d'une [invocation réentrante de l'analyseur](#). Si l'[indicateur de pause de l'analyseur](#) est défini, le tokenizer s'arrêtera immédiatement et aucun code HTML ne sera analysé, conformément à la [vérification de l'indicateur de pause de l'analyseur](#) du tokenizer.*

Les `document.write(...)` étapes de la méthode consistent à exécuter les [étapes d'écriture du document](#) avec [this](#) et une chaîne qui est la concaténation de tous les arguments passés.

#### 8.4.4 `document.writeln()`

`document.writeln(...text)`



Ajoute la ou les chaînes données au [Document](#) flux d'entrée de , suivies d'un caractère de saut de ligne. Si nécessaire, appelle `open()` implicitement la méthode en premier.

Lance un `"_InvalidStateError" DOMException` lorsqu'il est appelé sur [des documents XML](#).

Lève un `"_InvalidStateError" DOMException` si l'analyseur est en train d'exécuter un [constructeur d'élément personnalisé](#).



Les `document.writeln(...)` étapes de la méthode consistent à exécuter les [étapes d'écriture du document](#) avec [this](#) et une chaîne qui est la concaténation de tous les arguments passés et U+000A LINE FEED.

## 8.5 Analyse DOM



L' `DOMParser` interface permet aux auteurs de créer de nouveaux `Document` objets en analysant des chaînes, au format HTML ou XML.

```
parser = new DOMParser()
```



Construit un nouvel `DOMParser` objet.

```
document = parser.parseFromString(string, type)
```



Analyse *la chaîne* à l'aide de l'analyseur HTML ou XML, selon *le type*, et renvoie le résultat `Document`. *type* peut être "`text/html`" (qui appellera l'analyseur HTML), ou l'un des "`text/xml`", "`application/xml`", "`application/xhtml+xml`" ou "`image/svg+xml`" (qui appellera l'analyseur XML).

Pour l'analyseur XML, si *la chaîne* ne peut pas être analysée, le retour `Document` contiendra des éléments décrivant l'erreur résultante.

Notez que `script` les éléments ne sont pas évalués lors de l'analyse et que l' [encodage](#) du document résultant sera toujours [UTF-8](#).

Les valeurs autres que celles ci-dessus pour *le type* entraîneront `TypeError` la levée d'une exception.

*La conception de `DOMParser`, en tant que classe qui doit être construite puis appelée sa `parseFromString()` méthode, est un artefact historique malheureux. Si nous devions concevoir cette fonctionnalité aujourd'hui, ce serait une fonction autonome.*

```
[Exposed=Window]
```

```
interface DOMParser {
```

```
  constructor();
```

```
[NewObject] Document parseFromString(DOMString string,
```

```
  DOMParserSupportedType type);
```

```
};
```

```
enum DOMParserSupportedType {
```

```
"text/html",
```

```
"text/xml",
```

```
"application/xml",
```

```
"application/xhtml+xml",
```

```
"image/svg+xml"
```

```
};
```

Les `new DOMParser()` étapes du constructeur sont de ne rien faire.

Les étapes de la méthode sont : `parseFromString(string, type)`

1. Soit *document* un `new Document`, dont le type de contenu est `type` et url est l'URL associée à cet objet global pertinent `.Document`

*L'encodage du document sera laissé par défaut, UTF-8. En particulier, toutes les déclarations ou metaéléments XML trouvés lors de l'analyse de la chaîne n'auront aucun effet.*

2. Allumer le `type` :

`"text/html"`

1. Définissez le type de *document* sur `".html"`
2. Créez un parseur HTML parser, associé à *document*.
3. Placez la chaîne dans le flux d'entrée pour l'analyseur. La confiance d'encodage n'est pas pertinente.
4. Démarrez l'analyseur et laissez-le s'exécuter jusqu'à ce qu'il ait consommé tous les caractères qui viennent d'être insérés dans le flux d'entrée.

*Cela peut modifier le mode du document.*

*Puisque le document n'a pas de contexte de navigation, le script est désactivé.*

**Sinon**

5. Créez un [analyseur XML](#) *parse* , associé au *document* , et avec [la prise en charge des scripts XML désactivée](#) .
6. Analyser *la chaîne* à l'aide de *parser* .
7. Si l'étape précédente a entraîné une erreur de bonne formation XML ou d'espace de noms XML, alors :
  1. [Assert](#) : *le document* n'a pas de nœuds enfants.
  2. Soit *root* le résultat de [la création d'un élément](#) donné *document* , " `parsererror` " et " `http://www.mozilla.org/newlayout/xml/parsererror.xml` ".
  3. Ajoutez éventuellement des attributs ou des enfants à *la racine* pour décrire la nature de l'erreur d'analyse.
  4. [Ajouter](#) *la racine* au *document* .
3. *Document* de retour .

[← 8.4 Insertion de balisage dynamique](#) — [Table des matières](#) — [8.9 État et capacités du système](#) →

1.

1. [8.6 Minuteries](#)
2. [8.7 Mise en file d'attente des microtâches](#)
3. [8.8 Invites de l'utilisateur](#)
  1. [8.8.1 Boîtes de dialogue simples](#)
  2. [8.8.2 Impression](#)

## 8.6 Minuteries

Les méthodes `setTimeout()` et `setInterval()` permettent aux auteurs de planifier des rappels basés sur un minuteur.

```
id = self.setTimeout(handler [, timeout [, ...arguments ] ])
```

✓

Planifie un délai d'attente pour exécuter *le gestionnaire* après *un délai d'attente* de millisecondes. Tous *les arguments* sont transmis directement au *gestionnaire* .

```
id = self.setTimeout(code [, timeout ])
```

Planifie un délai d'attente pour compiler et exécuter *le code* après *le délai d'attente* millisecondes.

```
self.clearTimeout(id)
```

✓

Annule le délai défini avec `setTimeout()` ou `setInterval()` identifié par *id*.

```
id = self.setInterval(handler [, timeout [, ...arguments ] ])
```

✓

Planifie un délai d'attente pour exécuter *le gestionnaire* toutes les millisecondes de *délai d'attente*. Tous les *arguments* sont transmis directement au *gestionnaire*.

```
id = self.setInterval(code [, timeout ])
```

Planifie un délai d'attente pour compiler et exécuter *du code* toutes les millisecondes de *délai d'attente*.

```
self.clearInterval(id)
```

✓

Annule le délai défini avec `setInterval()` ou `setTimeout()` identifié par *id*.

*Les minuteries peuvent être imbriquées ; après cinq de ces temporisateurs imbriqués, cependant, l'intervalle est forcé à être d'au moins quatre millisecondes. Cette API ne garantit pas que les temporisateurs s'exécuteront exactement dans les délais. Des retards dus à la charge du processeur, à d'autres tâches, etc. sont à prévoir.*

Les objets qui implémentent le `WindowOrWorkerGlobalScope` mixin ont une **map of active timers**, qui est une [map](#), initialement vide. Chaque [clé](#) de cette carte est un identifiant pour une minuterie, et chaque [valeur](#) est un `DOMHighResTimeStamp`, représentant l'heure d'expiration de cette minuterie.

*Pour les entrées placées dans la [carte des temporisateurs actifs](#) par les [étapes d'initialisation des temporisateurs](#), c'est-à-dire par `setTimeout()` et `setInterval()`, les clés sont des nombres. Pour les autres spécifications qui utilisent les [étapes d'exécution après un](#) algorithme de temporisation, l'identificateur est une valeur non numérique unique. Seuls les temporisateurs à touches numériques sont affectés par `clearTimeout()` et `clearInterval()`, mais tous les temporisateurs contribuent au [calcul du délai d'inactivité](#) et sont effacés lorsque le global concerné est détruit.*

---

Les étapes de la méthode consistent à renvoyer le résultat de l'exécution des [étapes d'initialisation du minuteur](#) étant donné [this](#), *handler*, *timeout*, *arguments* et `false.setTimeout(handler, timeout, ...arguments)`

Les étapes de la méthode consistent à renvoyer le résultat de l'exécution des [étapes d'initialisation du minuteur](#) étant donné [this](#) , *handler* , *timeout* , *arguments* et `true.setInterval (handler , timeout , ...arguments)`

Les étapes de la méthode et consistent à [supprimer cette](#) carte [des minuteurs actifs](#) [ *id* ].`clearTimeout (id)` `clearInterval (id)`

*Étant donné que `clearTimeout()` et `clearInterval()` effacez les entrées de la même carte, l'une ou l'autre méthode peut être utilisée pour effacer les minuteries créées par `setTimeout()` ou `setInterval()`.*

---

Les **étapes d'initialisation de la minuterie** , étant donné un [WindowOrWorkerGlobalScope](#) *global* , une chaîne ou [Function](#) *un gestionnaire* , un nombre *timeout* , une liste *arguments* , un booléen *repeat* , et éventuellement (et uniquement si *repeat* est vrai) un nombre *previousId* , sont :

1. Soit *thisArg* global si c'est un objet ; [WorkerGlobalScope](#) sinon, laissez *thisArg* être celui [WindowProxy](#) qui correspond à *global* .
2. Si *previousId* a été donné, soit *id* soit *previousId* ; sinon, laissez *id* être un entier [défini par l'implémentation](#) qui est supérieur à zéro et n'existe pas déjà [dans](#) la carte globale des [temporiseurs actifs](#) .
3. Si la [tâche en cours d'exécution](#) de [la boucle d'événements](#) de l' [agent environnant](#) est une tâche qui a été créée par cet algorithme, laissez *le niveau d'imbrication* être le [niveau d'imbrication](#) du temporisateur de la [tâche](#) . Sinon, laissez *le niveau d'imbrication* être égal à zéro.

*Le [niveau d'imbrication des temporiseurs](#) de la tâche est utilisé à la fois pour les appels imbriqués à `setTimeout()` et pour les temporiseurs répétés créés par `setInterval()` . (Ou, en fait, pour toute combinaison des deux.) En d'autres termes, il représente des invocations imbriquées de cet algorithme, et non d'une méthode particulière.*

4. Si *le délai d'attente* est inférieur à 0, définissez *le délai d'attente* sur 0.
5. Si *le niveau d'imbrication* est supérieur à 5 et que *le délai d'attente* est inférieur à 4, définissez *le délai d'attente* sur 4.
6. Soit *domaine* le [domaine pertinent](#) de *global* .
7. Soit *le script d'initiation* le [script actif](#) .
8. [Assert](#) : *le script d'initiation* n'est pas nul, puisque cet algorithme est toujours appelé depuis un script.

9. Soit *tâche* une [tâche](#) qui exécute les sous-étapes suivantes :
1. Si *id* n'existe pas [dans](#) la [carte des minuteurs actifs](#) de *global* , annulez ces étapes.
  2. Si *handler* est un [Function](#), alors [appelez](#) le *gestionnaire* en lui donnant *des arguments* avec le [rappel this value](#) défini sur *thisArg* . Si cela lève une exception, attrapez-la et [signalez l'exception](#) .
  3. Sinon:
    1. [Assert](#) : le *gestionnaire* est une chaîne.
    2. Effectuez [HostEnsureCanCompileStrings](#) ( *domaine* ). Si cela lève une exception, interceptez-la, [signalez l'exception](#) et abandonnez ces étapes.
    3. Soit *settings object* l' objet de [paramètres pertinent](#) *global* .
    4. Laissez l' URL de base être [l' URL de base](#) du *script initial* .
    5. [Assert](#) : l'URL de base n'est pas nulle, car le *script d'initiation* est un [script classique](#) ou un [script de module JavaScript](#) .
    6. Soit *les options de récupération* d'un [script fetch options](#) dont le [nonce cryptographique](#) lance le [nonce cryptographique](#) des [options de récupération du script](#) , les [métadonnées d'intégrité](#) sont la chaîne vide, [les métadonnées de l'analyseur](#) sont " " , [le mode d'identification](#) lance le mode d'identification des [options de récupération](#) du *script* , [la politique de référence](#) lance [la politique de référence](#) des [options de récupération](#) du *script* , et `not-parser-inserted` [la priorité de récupération](#) est " auto".
- L'effet de ces options garantit que la compilation de chaînes effectuée par `setTimeout()` et `setInterval()` se comporte de manière équivalente à celle effectuée par `eval()` . Autrement dit, les extractions [de script de module](#) via `import()` se comporteront de la même manière dans les deux contextes.*
7. Soit *script* le résultat de [la création d'un script classique](#) avec un *gestionnaire* , un *objet de paramètres* , une *URL de base* et des *options de récupération* .
  8. [Exécutez le](#) *script* de *script classique* .
4. Si *id* n'existe pas [dans](#) la [carte des minuteurs actifs](#) de *global* , annulez ces étapes.

*Il a peut-être été supprimé via le code de l'auteur dans l'appel du *gestionnaire* `clearTimeout()` ou `clearInterval()`.*

5. Si *repeat* vaut true, effectuez à nouveau les [étapes d'initialisation de la minuterie](#) , étant donné *global* , *handler* , *timeout* , *arguments* , true et *id* .
6. Sinon, [supprimez](#) la carte globale [des minuteurs actifs](#) [ *id* ].
10. Incrémente le niveau d'imbrication de un.
11. Définissez le niveau d'imbrication du minuteur de la tâche sur le niveau d'imbrication .
12. Soit *completeStep* une étape d'algorithme qui [met en file d'attente une tâche globale](#) sur la **source de la tâche du minuteur** donnée *global* pour exécuter la tâche .
13. [Exécutez les étapes après un délai d'expiration](#) donné *global* , " *setTimeout/setInterval* ", *timeout* , *CompletionStep* et *id* .
14. *Identifiant* de retour .

*La conversion d'argument telle que définie par Web IDL (par exemple, invoquer `toString()` des méthodes sur des objets passés comme premier argument) se produit dans les algorithmes définis dans Web IDL, avant que cet algorithme ne soit invoqué.*

Ainsi, par exemple, le code plutôt idiot suivant se traduira par le journal contenant " ONE TWO " :

```
var log = '';  
function logger(s) { log += s + ' '; }  
  
setTimeout({ toString: function () {  
    setTimeout("logger('ONE')", 100);  
    return "logger('TWO')";  
} }, 100);
```

Pour exécuter des tâches de plusieurs millisecondes consécutives sans délai, tout en cédant au navigateur pour éviter d'affamer l'interface utilisateur (et pour éviter que le navigateur ne tue le script pour monopoliser le processeur), mettez simplement le prochain minuteur en file d'attente avant d'effectuer le travail :

```
function doExpensiveWork() {  
    var done = false;  
    // ...  
    // this part of the function takes up to five milliseconds  
    // set done to true if we're done  
    // ...  
    return done;  
}
```

```

function rescheduleWork() {
    var id = setTimeout(rescheduleWork, 0); // preschedule next
    iteration
    if (doExpensiveWork())
        clearTimeout(id); // clear the timeout if we don't need it
}

function scheduleWork() {
    setTimeout(rescheduleWork, 0);
}

scheduleWork(); // queues a task to do lots of work

```

Pour **exécuter des étapes après un délai d'expiration** , étant donné un [WindowOrWorkerGlobalScope](#) *global* , une chaîne *orderingIdentifieur* , un nombre de *millisecondes* , un ensemble d'étapes *CompletionSteps* et une valeur facultative *timerKey* :

1. [Assert](#) : si *timerKey* est donné, alors l'appelant de cet algorithme est le [timer initialization steps](#) . (Les autres spécifications ne doivent pas passer *timerKey* .)
2. Si *timerKey* n'est pas donné, définissez-le sur une nouvelle valeur non numérique unique.
3. Soit *startTime* l' [heure haute résolution actuelle](#) donnée *global* .
4. [Définissez](#) la carte *globale des temporisateurs actifs* [ *timerKey* ] sur *startTime* plus les *millisecondes* .
5. Exécutez les étapes suivantes [en parallèle](#) :
  1. Si *global* est un [Window](#) objet, attendez que l'[associé](#) de *global* soit [pleinement actif](#) pendant encore quelques *millisecondes* millisecondes (pas nécessairement consécutives). [Document](#)
  - Sinon, *global* est un [WorkerGlobalScope](#) objet ; attendez que *millisecondes* millisecondes se soient écoulées sans que le travailleur ne soit suspendu (pas nécessairement consécutivement).
  2. Attendez que toutes les invocations de cet algorithme qui avaient le même *global* et *orderingIdentifieur* , qui ont commencé avant celui-ci, et dont les *millisecondes* sont égales ou inférieures à celle-ci, se soient terminées.
  3. Facultativement, attendez une durée supplémentaire [définie par l'implémentation](#) .



*Ceci est destiné à permettre aux agents utilisateurs de compléter les délais d'attente selon les besoins pour optimiser la consommation d'énergie de l'appareil. Par exemple, certains processeurs ont un mode basse consommation où la granularité des temporisateurs est réduite ; sur de telles plates-formes, les agents utilisateurs peuvent ralentir les temporisateurs pour s'adapter à ce calendrier au lieu d'exiger que le processeur utilise le mode le plus précis avec sa consommation d'énergie plus élevée associée.*

4. Effectuez les étapes d'achèvement .
5. Si `timerKey` est une valeur non numérique, [supprimez](#) la carte globale [des minuteurs actifs](#) [ `timerKey` ].

[Les étapes d'exécution après un délai d'expiration](#) sont destinées à être utilisées par d'autres spécifications qui souhaitent exécuter du code fourni par le développeur après un délai d'expiration fourni par le développeur, de la même manière que `setTimeout()`. (Notez, cependant, qu'il n'a pas le comportement d'imbrication et de verrouillage de `setTimeout()`.) De telles spécifications peuvent choisir un `orderingIdentifier` pour assurer la commande dans les délais d'attente de leur spécification, sans contraindre la commande par rapport aux délais d'attente d'autres spécifications.

## 8.7 Mise en file d'attente des microtâches



`self.queueMicrotask(callback)`

[Met en file d'attente](#) une [microtâche](#) pour exécuter le *rappel* donné .

La méthode doit [mettre en file d'attente une microtâche](#) pour [invoquer](#) `callback` , et si `callback` lève une exception, [signaler l'exception](#) `.queueMicrotask(callback)`

La `queueMicrotask()` méthode permet aux auteurs de programmer un rappel sur la [file d'attente des microtâches](#) . Cela permet à leur code de s'exécuter une fois que la [pile de contexte d'exécution JavaScript](#) est ensuite vide, ce qui se produit une fois que tout le JavaScript synchrone en cours d'exécution est terminé. Cela ne rend pas le contrôle à la [boucle d'événements](#) , comme ce serait le cas lors de l'utilisation, par exemple, de `.setTimeout(f, 0)`

Les auteurs doivent être conscients que la planification d'un grand nombre de microtâches présente les mêmes inconvénients en termes de performances que l'exécution d'un grand nombre de codes synchrones. Les deux empêcheront le navigateur de faire son propre travail, comme le rendu. Dans de nombreux cas, `requestAnimationFrame()` ou `requestIdleCallback()` est un meilleur choix. En particulier, si l'objectif est d'exécuter du code avant le prochain cycle de rendu, c'est le but de `requestAnimationFrame()`.

Comme on peut le voir dans les exemples suivants, la meilleure façon de penser `queueMicrotask()` est d'utiliser un mécanisme de réorganisation du code synchrone, en plaçant effectivement le code en file d'attente immédiatement après l'exécution du JavaScript synchrone en cours d'exécution.

La raison la plus courante d'utilisation `queueMicrotask()` est de créer un classement cohérent, même dans les cas où les informations sont disponibles de manière synchrone, sans introduire de retard indu.

Par exemple, considérez un élément personnalisé déclenchant un `load` événement, qui maintient également un cache interne des données précédemment chargées. Une implémentation naïve pourrait ressembler à :

```
MyElement.prototype.loadData = function (url) {
  if (this._cache[url]) {
    this._setData(this._cache[url]);
    this.dispatchEvent(new Event("load"));
  } else {
    fetch(url).then(res => res.arrayBuffer()).then(data => {
      this._cache[url] = data;
      this._setData(data);
      this.dispatchEvent(new Event("load"));
    });
  }
};
```

Cette implémentation naïve est cependant problématique en ce sens qu'elle provoque chez ses utilisateurs un comportement incohérent. Par exemple, un code tel que

```
element.addEventListener("load", () => console.log("loaded"));
console.log("1");
element.loadData();
console.log("2");
```

enregistrera parfois "1, 2, chargé" (si les données doivent être récupérées), et parfois enregistrera "1, chargé, 2" (si les données sont déjà en cache). De même, après l'appel à `loadData()`, il sera incohérent que les données soient définies ou non sur l'élément.

Pour obtenir une commande cohérente, `queueMicrotask()` peut être utilisé :

```
MyElement.prototype.loadData = function (url) {
  if (this._cache[url]) {
    queueMicrotask(() => {
      this._setData(this._cache[url]);
    });
  }
};
```

```

    this.dispatchEvent(new Event("load"));
  });
} else {
  fetch(url).then(res => res.arrayBuffer()).then(data => {
    this._cache[url] = data;
    this._setData(data);
    this.dispatchEvent(new Event("load"));
  });
}
};

```

En réorganisant essentiellement le code en file d'attente après que la [pile de contexte d'exécution JavaScript](#) se vide, cela garantit un ordre et une mise à jour cohérents de l'état de l'élément.

Une autre utilisation intéressante de [queueMicrotask\(\)](#) est de permettre un "lot" non coordonné de travail par plusieurs appelants. Par exemple, considérez une fonction de bibliothèque qui souhaite envoyer des données quelque part dès que possible, mais ne souhaite pas effectuer plusieurs requêtes réseau si cela est facilement évitable. Une façon d'équilibrer cela serait comme ceci:

```

const queuedToSend = [];

function sendData(data) {
  queuedToSend.push(data);

  if (queuedToSend.length === 1) {
    queueMicrotask(() => {
      const stringToSend = JSON.stringify(queuedToSend);
      queuedToSend.length = 0;

      fetch("/endpoint", stringToSend);
    });
  }
}

```

Avec cette architecture, plusieurs appels ultérieurs à `sendData()` l'intérieur du JavaScript synchrone en cours d'exécution seront regroupés en un seul [fetch\(\)](#) appel, mais sans aucune tâche de boucle d'événement intermédiaire préemptant la récupération (comme cela se serait produit avec un code similaire qui utilisait à la place [setTimeout\(\)](#)).

## 8.8 Invites de l'utilisateur

### 8.8.1 Boîtes de dialogue simples

```
window.alert(message)
```

✓

Affiche une alerte modale avec le message donné et attend que l'utilisateur la rejette.

```
result = window.confirm(message)
```

✓

Affiche une invite modale OK/Annuler avec le message donné, attend que l'utilisateur le rejette et renvoie vrai si l'utilisateur clique sur OK et faux s'il clique sur Annuler.

```
result = window.prompt(message [, default])
```

✓

Affiche une invite de contrôle de texte modal avec le message donné, attend que l'utilisateur le rejette et renvoie la valeur que l'utilisateur a entrée. Si l'utilisateur annule l'invite, renvoie null à la place. Si le deuxième argument est présent, alors la valeur donnée est utilisée par défaut.

*La logique qui dépend des [tâches](#) ou [des microtâches](#), telles que [les éléments multimédias](#) chargeant leurs [données multimédias](#), est bloquée lorsque ces méthodes sont invoquées.*

Les étapes de la méthode `alert()` et sont :`alert(message)`

1. Si nous [ne pouvons pas afficher de boîtes de dialogue simples](#) pour [cela](#), alors revenez.
2. Si la méthode a été invoquée sans arguments, laissez *message* être la chaîne vide ; sinon, laissez *message* être le premier argument de la méthode.
3. Définissez *message* sur le résultat de [la normalisation des sauts de ligne](#) donné *message*.
4. Définissez *message* sur le résultat de [la troncation facultative](#) de *message*.
5. Afficher *le message* à l'utilisateur, traitant U+000A LF comme un saut de ligne.
6. Appelez [l'invite utilisateur WebDriver BiDi ouverte](#) avec [ceci](#), " `alert`" et *le message*.
7. Si vous le souhaitez, [faites une pause](#) en attendant que l'utilisateur accuse réception du message.
8. Appelez [l'invite utilisateur WebDriver BiDi fermée](#) avec [ceci](#) et vrai.

Cette méthode est définie à l'aide de deux surcharges, au lieu d'utiliser un argument facultatif, pour des raisons historiques. L'impact pratique de ceci est que `alert(undefined)` est traité comme `alert("undefined")`, mais `alert()` est traité comme `alert("")`.

Les étapes de la méthode sont :`confirm(message)`

1. Si nous ne pouvons pas afficher de boîtes de dialogue simples pour cela , renvoyez false.
2. Définissez *message* sur le résultat de la normalisation des sauts de ligne donné *message* .
3. Définissez *message* sur le résultat de la troncation facultative de *message* .
4. Afficher *le message* à l'utilisateur, en traitant U+000A LF comme un saut de ligne, et demander à l'utilisateur de répondre par une réponse positive ou négative.
5. Appelez l'invite utilisateur WebDriver BiDi ouverte avec ceci , " confirm" et *le message* .
6. Faites une pause jusqu'à ce que l'utilisateur réponde positivement ou négativement.
7. Appelez l'invite utilisateur WebDriver BiDi fermée avec this , et true si l'utilisateur a répondu positivement ou false dans le cas contraire.
8. Si l'utilisateur a répondu positivement, retourne true ; sinon, l'utilisateur a répondu par la négative : return false.

Les étapes de la méthode sont :`prompt(message , default)`

1. Si nous ne pouvons pas afficher de boîtes de dialogue simples pour this , renvoyez null.
2. Définissez *message* sur le résultat de la normalisation des sauts de ligne donné *message* .
3. Définissez *message* sur le résultat de la troncation facultative de *message* .
4. Définissez *default* sur le résultat de la troncation facultative de *default* .
5. Afficher *le message* à l'utilisateur, en traitant U+000A LF comme un saut de ligne, et demander à l'utilisateur de répondre avec une valeur de chaîne ou d'abandonner. La réponse doit être définie par défaut sur la valeur donnée par *défaut* .
6. Appelez l'invite utilisateur WebDriver BiDi ouverte avec ceci , " prompt" et *le message* .

7. [Faites une pause](#) en attendant la réponse de l'utilisateur.
8. Laissez *result* être null si l'utilisateur abandonne, ou sinon la chaîne avec laquelle l'utilisateur a répondu.
9. Appelez [l'invite utilisateur WebDriver BiDi fermée](#) avec [this](#) , false si *le résultat* est null ou true sinon, et *result* .
10. Retourner *le résultat* .

Pour **éventuellement tronquer une chaîne de dialogue simple** *s* , renvoyez soit *s* lui-même, soit une chaîne dérivée de *s* plus courte. Les agents utilisateurs ne doivent pas fournir d'interface utilisateur pour afficher la partie éliminée de *s* , car cela rend trop facile pour les abuseurs de créer des boîtes de dialogue de la forme "Importante alerte de sécurité ! Cliquez sur 'Afficher plus' pour plus de détails !".

*Par exemple, un agent utilisateur peut souhaiter n'afficher que les 100 premiers caractères d'un message. Ou, un agent utilisateur peut remplacer le milieu de la chaîne par "...". Ces types de modifications peuvent être utiles pour limiter le potentiel d'abus de boîtes de dialogue système anormalement grandes et dignes de confiance.*

Nous **ne pouvons pas afficher de boîtes de dialogue simples** pour une [Window](#) *fenêtre* lorsque l'algorithme suivant renvoie true :

1. Si l' [ensemble d'indicateurs de sandboxing](#) [actif](#) de la *fenêtre associée* [Document](#) a l' [indicateur de mode sandbox](#) défini, alors renvoie true.
2. Si l'[origine](#) de l'[objet de paramètres pertinent](#) de la *fenêtre* et l'[origine de niveau supérieur](#) de l'[objet de paramètres pertinent](#) de la *fenêtre* ne sont pas [le même domaine d'origine](#) , alors renvoie true.
3. Si [le niveau d'imbrication de terminaison](#) de [la boucle d'événements](#) de l'[agent concerné](#) de la *fenêtre* est différent de zéro, alors éventuellement renvoyer true.
4. En option, retournez true. (Par exemple, l'agent utilisateur pourrait donner à l'utilisateur la possibilité d'ignorer tous les dialogues modaux, et abandonnerait ainsi à cette étape chaque fois que la méthode était invoquée.)
5. Renvoie faux.

## 8.8.2 Impression



```
window.print()
```

Invite l'utilisateur à imprimer la page.

Les `print()` étapes de la méthode sont :

1. Soit le document associé à ceci `.Document`
2. Si le document n'est pas complètement actif , alors revenez.
3. Si le compteur de déchargement du document est supérieur à 0, alors retour.
4. Si le document est prêt pour les tâches de post-chargement , exécutez les étapes d'impression pour le document .
5. Sinon, définissez l'indicateur d'impression du document **lors du chargement** .

Les agents utilisateurs doivent également exécuter les étapes d'impression chaque fois que l'utilisateur demande l'opportunité d' obtenir une forme physique (par exemple une copie imprimée), ou la représentation d'une forme physique (par exemple une copie PDF), d'un document.

Les **étapes d'impression** d'un `Document` document sont :

1. L'agent utilisateur peut afficher un message à l'utilisateur ou revenir (ou les deux).

Par exemple, un navigateur de kiosque pourrait ignorer silencieusement toutes les invocations de la `print()` méthode.

Par exemple, un navigateur sur un appareil mobile pourrait détecter qu'il n'y a pas d'imprimantes à proximité et afficher un message le disant avant de continuer à proposer une option "enregistrer au format PDF".

2. Si l' ensemble d'indicateurs sandboxing actif du document a l' indicateur modal sandbox défini, alors revenez.

*Si la boîte de dialogue d'impression est bloquée par un `Document` bac à sable de , alors ni les événements `beforeprint` ni ne `afterprint` seront déclenchés.*

3. L'agent utilisateur doit déclencher un événement nommé `beforeprint` sur l' objet global pertinent de document , ainsi que sur tout enfant navigable dans celui-ci.

**Tirer chez les enfants ne semble pas juste ici, et certaines tâches doivent probablement être mises en file d'attente. Voir [numéro 5096](#) .**

L' `beforeprint` événement peut être utilisé pour annoter la copie imprimée, par exemple en ajoutant l'heure à laquelle le document a été imprimé.

4. L'agent utilisateur doit offrir à l'utilisateur la possibilité d' obtenir une forme physique (ou la représentation d'une forme physique) de document . L'agent utilisateur peut attendre que l'utilisateur accepte ou refuse avant de revenir ; si c'est le cas, l'agent utilisateur doit faire une pause pendant que la méthode

attend. Même si l'agent utilisateur n'attend pas à ce stade, l'agent utilisateur doit utiliser l'état des documents pertinents tels qu'ils sont à ce stade de l'algorithme si et quand il crée éventuellement le formulaire alternatif.

5. L'agent utilisateur doit [déclencher un événement](#) nommé `afterprint` sur l' [objet global pertinent](#) de `document` , ainsi que sur tous les [objets navigables enfants](#) qu'il contient.

Tirer chez les enfants ne semble pas juste ici, et certaines tâches doivent probablement être mises en file d'attente. Voir [numéro 5096](#) .

L' `afterprint` événement peut être utilisé pour annuler les annotations ajoutées dans l'événement précédent, ainsi que pour afficher l'interface utilisateur post-impression. Par exemple, si une page guide l'utilisateur à travers les étapes d'une demande de prêt immobilier, le script pourrait passer automatiquement à l'étape suivante après avoir imprimé un formulaire ou autre.

1.

## 1. [8.9 État et capacités du système](#)

### 1. [8.9.1 L' `Navigator` objet](#)

1. [8.9.1.1 Identification du client](#)
2. [8.9.1.2 Préférences de langue](#)
3. [8.9.1.3 État du navigateur](#)
4. [8.9.1.4 Gestionnaires de schémas personnalisés :](#)  
[la `registerProtocolHandler\(\)` méthode](#)
  1. [8.9.1.4.1 Sécurité et confidentialité](#)
  2. [8.9.1.4.2 Automatisation de l'agent utilisateur](#)
5. [8.9.1.5 Cookies](#)
6. [8.9.1.6 Prise en charge de l'affichage PDF](#)

## 8.9 État et capacités du système

### 8.9.1 L' [Navigator](#) objet



Les instances de `Navigator` représentent l'identité et l'état de l'agent utilisateur (le client). Ils servent également de global générique sous lequel diverses API sont situées dans cette spécification et d'autres.

[Exposed=Window]



```
interface Navigator {
```

```
// objects implementing this interface also implement the
```

```
interfaces given below
```

```
};
```

```
Navigator includes NavigatorID;
```

```
Navigator includes NavigatorLanguage;
```

```
Navigator includes NavigatorOnLine;
```

```
Navigator includes NavigatorContentUtils;
```

```
Navigator includes NavigatorCookies;
```

```
Navigator includes NavigatorPlugins;
```

```
Navigator includes NavigatorConcurrentHardware;
```

*Ces mixins d'interface sont définis séparément afin de [WorkerNavigator](#) pouvoir réutiliser des parties de l' [Navigator](#) interface.*

Chacun [Window](#) a un **associé** [Navigator](#) , qui est un [Navigator](#) objet. Lors de la création de l' [Window](#) objet, son **associé** [Navigator](#) doit être défini sur un **nouvel** [Navigator](#) objet créé dans le **domaine pertinent** [Window](#) de l'objet .

✓ MDN

Les étapes `getter navigator` et `clientInformation` getter consistent à retourner [this](#) 's [relatedNavigator](#) .

### 8.9.1.1 Identification du client

```
interface mixin NavigatorID {
```

```
  readonly attribute DOMString appCodeName; // constant "Mozilla"
```

```
  readonly attribute DOMString appName; // constant "Netscape"
```

```
  readonly attribute DOMString appVersion;
```

```
readonly attribute DOMString platform;
```

```
readonly attribute DOMString product; // constant "Gecko"
```

```
[Exposed=Window] readonly attribute DOMString productSub;
```

```
readonly attribute DOMString userAgent;
```

```
[Exposed=Window] readonly attribute DOMString vendor;
```

```
[Exposed=Window] readonly attribute DOMString vendorSub; //
```

```
constant ""
```

```
};
```

Dans certains cas, malgré les meilleurs efforts de l'ensemble de l'industrie, les navigateurs Web présentent des bogues et des limitations que les auteurs Web sont obligés de contourner.

Cette section définit une collection d'attributs qui peuvent être utilisés pour déterminer, à partir du script, le type d'agent utilisateur utilisé, afin de contourner ces problèmes.

L'agent utilisateur a un **mode de compatibilité de navigateur**, qui est *Chrome*, *Gecko* ou *WebKit*.

*Le [mode de compatibilité du navigateur](#) contraint le [NavigatorID](#) mixin aux combinaisons de valeurs d'attribut et de présence de [taintEnabled\(\)](#) et [oscpu](#) qui sont connues pour être compatibles avec le contenu Web existant.*

La détection des clients doit toujours être limitée à la détection des versions actuelles connues ; les versions futures et les versions inconnues doivent toujours être considérées comme entièrement conformes.

[self.navigator.appCodeName](#)

Renvoie la chaîne "Mozilla".

[self.navigator.appName](#)

Renvoie la chaîne "Netscape".

[self.navigator.appVersion](#)

Renvoie la version du navigateur.

[self.navigator.platform](#)

Renvoie le nom de la plateforme.

[self.navigator.product](#)

Renvoie la chaîne "Gecko".

window.navigator.productSub

Renvoie soit la chaîne " 20030107", soit la chaîne " 20100101".

self.navigator.userAgent

✓

Renvoie l' User-Agent en-tête `` complet.

window.navigator.vendor

Renvoie soit la chaîne vide, soit la chaîne " Apple Computer, Inc.", soit la chaîne " Google Inc.".

window.navigator.vendorSub

Renvoie la chaîne vide.

**appName**

Doit renvoyer la chaîne " Mozilla".

**appName**

Doit renvoyer la chaîne " Netscape".

**appVersion**

Doit renvoyer la chaîne appropriée qui commence par " 5.0 (" , comme suit :

Soit *trail* la sous-chaîne de [la valeur par défaut](#) `User-Agent` qui suit le Mozilla/préfixe " ".

Si le [mode de compatibilité du navigateur](#) est **Chrome** ou **WebKit**

*Piste de retour* .

Si le [mode de compatibilité du navigateur](#) est **Gecko**

Si *la piste* commence par " 5.0 (Windows", alors retourne " 5.0 (Windows)".

Sinon, renvoyez le préfixe de *chemin* jusqu'au premier U+003B (;) non compris, concaténé avec le caractère U+0029 PARENTHÈSE DROITE. Par exemple, " 5.0 (Macintosh)", " 5.0 (Android 10)" ou " 5.0 (X11)".

**platform**

Doit renvoyer une chaîne représentant la plate-forme sur laquelle le navigateur s'exécute (par exemple " " MacIntel, " Win32", " Linux x86\_64", " Linux armv81") ou, pour des raisons de confidentialité et de compatibilité, une chaîne qui est généralement renvoyée sur une autre plate-forme.

**product**

Doit renvoyer la chaîne " Gecko".

**productSub**

Doit renvoyer la chaîne appropriée dans la liste suivante :

Si le [mode de compatibilité du navigateur](#) est **Chrome** ou **WebKit**

La chaîne " 20030107".

Si le mode de compatibilité du navigateur est **Gecko**

La chaîne " 20100101".

#### **userAgent**

Doit retourner la valeur `` par défautUser-Agent .

#### **vendor**

Doit renvoyer la chaîne appropriée dans la liste suivante :

Si le mode de compatibilité du navigateur est **Chrome**

La chaîne " Google Inc.".

Si le mode de compatibilité du navigateur est **Gecko**

La chaîne vide.

Si le mode de compatibilité du navigateur est **WebKit**

La chaîne " Apple Computer, Inc.".

#### **vendorSub**

Doit renvoyer la chaîne vide.

Si le mode de compatibilité du navigateur est **Gecko** , alors l'agent utilisateur doit également prendre en charge l'interface partielle suivante :

```
partial interface mixin NavigatorID {
```

```
[Exposed=Window] boolean taintEnabled(); // constant false
```

```
[Exposed=Window] readonly attribute DOMString oscpu;
```

```
};
```

La **taintEnabled()** méthode doit retourner false.

Le **oscpu**getter de l'attribut doit renvoyer soit la chaîne vide, soit une chaîne représentant la plate-forme sur laquelle le navigateur s'exécute, par exemple " Windows NT 10.0; Win64; x64", " Linux x86\_64".

**Toute information de cette API qui varie d'un utilisateur à l'autre peut être utilisée pour profiler l'utilisateur. En fait, si suffisamment d'informations de ce type sont disponibles, un utilisateur peut en fait être identifié de manière unique. Pour cette raison, les implémenteurs d'agents utilisateurs sont fortement invités à inclure le moins d'informations possible dans cette API.**

### 8.9.1.2 Préférences de langue

```
interface mixin NavigatorLanguage {
```

```
  readonly attribute DOMString language;
```

```
  readonly attribute FrozenArray<DOMString> languages;
```

```
};
```

**self.navigator.language**



Renvoie une balise de langue représentant la langue préférée de l'utilisateur.

**self.navigator.languages**



Renvoie un tableau de balises de langue représentant les langues préférées de l'utilisateur, la langue préférée en premier.

La langue préférée est celle renvoyée par [navigator.language](#).

Un [languagechange](#) événement est déclenché sur l'objet [Window](#) ou [WorkerGlobalScope](#) lorsque la compréhension par l'agent utilisateur des langues préférées de l'utilisateur change.

**language**

Doit renvoyer une balise de langue BCP 47 valide représentant soit [une langue plausible](#), soit la langue préférée de l'utilisateur. [\[BCP47\]](#)

**languages**

Doit renvoyer un [tableau figé](#) de balises de langue BCP 47 valides représentant une ou plusieurs [langues plausibles](#) ou les langues préférées de l'utilisateur, classées par préférence avec la langue la plus préférée en premier. Le même objet doit être renvoyé jusqu'à ce que l'agent utilisateur ait besoin de renvoyer des valeurs différentes ou des valeurs dans un ordre différent. [\[BCP47\]](#)

Chaque fois que l'agent utilisateur doit faire en sorte que l' [navigator.languages](#) attribut d'un [Window](#) ou [WorkerGlobalScope](#) d'un objet *global* renvoie un nouvel ensemble de balises de langue, l'agent utilisateur doit mettre en [file d'attente une tâche globale](#) sur la [source de tâche de manipulation DOM](#) donnée *globale* pour [déclencher un événement](#) nommé [languagechange](#) global, et attendre jusqu'à ce que cela tâche commence à être exécutée avant de renvoyer une nouvelle valeur.

Pour déterminer **un langage plausible**, l'agent utilisateur doit garder à l'esprit ce qui suit :

- Toute information de cette API qui varie d'un utilisateur à l'autre peut être utilisée pour profiler ou identifier l'utilisateur.
- Si l'utilisateur n'utilise pas un service qui obscurcit le point d'origine de l'utilisateur (par exemple, le réseau d'anonymat Tor), alors la valeur qui est la moins susceptible de distinguer l'utilisateur des autres utilisateurs ayant des origines similaires (par exemple, du même bloc d'adresse IP) est la langue utilisée par la majorité de ces utilisateurs. [\[TOR\]](#)
- Si l'utilisateur utilise un service d'anonymisation, la valeur " en-US" est suggérée ; si tous les utilisateurs du service utilisent cette même valeur, cela réduit la possibilité de distinguer les utilisateurs les uns des autres.

Pour éviter d'introduire d'autres vecteurs d'empreintes digitales, les agents utilisateurs doivent utiliser la même liste pour les API définies dans cette fonction que pour l' Accept-Language en-tête HTTP ``.

### 8.9.1.3 État du navigateur

```
interface mixin NavigatorOnline {
```

```
  readonly attribute boolean online;
```

```
};
```

**self.navigator.online**



Renvoie false si l'agent utilisateur est définitivement hors ligne (déconnecté du réseau). Renvoie true si l'agent utilisateur est peut-être en ligne.

Les événements [online](#) et [offline](#) sont déclenchés lorsque la valeur de cet attribut change.

L' **online** attribut doit retourner false si l'agent utilisateur ne contacte pas le réseau lorsque l'utilisateur suit des liens ou lorsqu'un script demande une page distante (ou sait qu'une telle tentative échouerait), et doit retourner true sinon.

Lorsque la valeur qui serait renvoyée par l' [navigator.online](#) attribut de a [Window](#) ou [WorkerGlobalScope](#) *global* passe de true à false, l'agent utilisateur doit [mettre en file d'attente une tâche globale](#) sur la [source de tâche réseau](#) donnée *global* pour [déclencher un événement](#) nommé [offline](#) global .

D'un autre côté, lorsque la valeur qui serait renvoyée par l' `navigator.onLine` attribut de a `Window` ou `WorkerGlobalScope` *global* passe de false à true, l'agent utilisateur doit mettre en file d'attente une tâche globale sur la source de tâche réseau donnée *global* pour déclencher un événement nommé `online` sur l' objet `Window` or `.WorkerGlobalScope`

*Cet attribut est par nature peu fiable. Un ordinateur peut être connecté à un réseau sans avoir accès à Internet.*

Dans cet exemple, un indicateur est mis à jour lorsque le navigateur se connecte et se déconnecte.

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <title>Online status</title>
    <script>
      function updateIndicator() {
        document.getElementById('indicator').textContent =
navigator.onLine ? 'online' : 'offline';
      }
    </script>
  </head>
  <body onload="updateIndicator()" ononline="updateIndicator()"
onoffline="updateIndicator()">
    <p>The network is: <span id="indicator">(state unknown)</span></p>
  </body>
</html>
```

#### 8.9.1.4 Gestionnaires de schémas personnalisés : la `registerProtocolHandler()` méthode

MDN

```
interface mixin NavigatorContentUtils {
```

```
  [SecureContext] undefined registerProtocolHandler(DOMString  
scheme, USVString url);
```

```
  [SecureContext] undefined unregisterProtocolHandler(DOMString  
scheme, USVString url);
```

```
};
```

```
window.navigator.registerProtocolHandler(scheme, url)
```

Enregistre un gestionnaire pour *le schéma* à *l'url*. Par exemple, un service de messagerie téléphonique en ligne pourrait s'enregistrer en tant que gestionnaire du `sms:` système, de sorte que si l'utilisateur clique sur un tel lien, il a la possibilité d'utiliser ce site Web. [\[SMS\]](#)

La chaîne "`%s`" dans *url* est utilisée comme espace réservé pour indiquer où placer l'URL du contenu à gérer.

Lève un "`SecurityError`" `DOMException` si l'agent utilisateur bloque l'enregistrement (cela peut arriver si vous essayez de vous enregistrer en tant que gestionnaire pour "`http`", par exemple).

Lance un "`SyntaxError`" `DOMException` si la "`%s`" chaîne est manquante dans *l'url*.

```
window.navigator.unregisterProtocolHandler(scheme, url)
```

Désenregistre le gestionnaire donné par les arguments.

Lève un "`SecurityError`" `DOMException` si l'agent utilisateur bloque la désinscription (cela peut arriver avec des schémas invalides, par exemple).

Lance un "`SyntaxError`" `DOMException` si la "`%s`" chaîne est manquante dans *l'url*.

Les étapes de la méthode sont : `registerProtocolHandler(scheme, url)`

1. Soit ( *normalizedScheme* , *normalizedURLString* ) le résultat de l'exécution [des paramètres de normalisation du gestionnaire de protocole](#) avec *le schéma* , *l'url* et [cet objet de paramètres](#) pertinent .
2. [En parallèle](#) : **enregistrez un gestionnaire de protocole** pour *normalizedScheme* et *normalizedURLString* . Les agents utilisateurs peuvent, dans les limites décrites, faire ce qu'ils veulent. Un agent utilisateur pourrait, par exemple, demander à l'utilisateur et offrir à l'utilisateur la possibilité d'ajouter le site à une liste restreinte de gestionnaires, ou de définir les gestionnaires par défaut, ou d'annuler la demande. Les agents utilisateurs pourraient également collecter silencieusement les informations, ne les fournissant que lorsqu'elles sont pertinentes pour l'utilisateur.

Les agents utilisateurs doivent garder une trace des sites qui ont enregistré des gestionnaires (même si l'utilisateur a refusé ces enregistrements) afin que l'utilisateur ne reçoive pas à plusieurs reprises la même demande.

Si le `registerProtocolHandler()` [mode d'automatisation](#) de [cet](#) objet [global pertinent](#) associé `n'estDocument` pas "`none`", l'agent utilisateur doit d'abord vérifier qu'il se trouve dans un contexte d'automatisation (voir [les considérations de sécurité de WebDriver](#) ). L'agent utilisateur doit alors contourner la communication d'informations ci-dessus et la collecte du consentement de l'utilisateur, et à la place faire ce qui suit en fonction de la valeur du `registerProtocolHandler()` [mode d'automatisation](#) :

" `auto-accept` "



Faites comme si l'utilisateur avait vu les détails de l'enregistrement et accepté la demande.

" **auto-reject** "

Faites comme si l'utilisateur avait vu les détails de l'inscription et avait rejeté la demande.

Lorsque l' **agent utilisateur utilise ce gestionnaire** pour une [URL](#) *inputURL* :

1. [Assert](#) : le [schéma](#) de *inputURL* est *normalizedScheme* .
2. [Définissez le nom d'utilisateur](#) donné *inputURL* et la chaîne vide.
3. [Définissez le mot de passe](#) donné *inputURL* et la chaîne vide.
4. Soit *inputURLString* la [sérialisation](#) de *inputURL* .
5. Soit *encodedURL* le résultat de l'exécution de l' [encodage de pourcentage UTF-8](#) sur *inputURLString* à l'aide du [composant percent-encode set](#) .
6. Laissez *handlerURLString* être *normalizedURLString* .
7. Remplacez la première instance de " %s " dans *handlerURLString* par *encodedURL* .
8. Soit *resultURL* le résultat de [l'analyse](#) de *handlerURLString* .
9. [Accédez à un élément navigable](#) approprié vers *resultURL* .

Si l'utilisateur avait visité un site à <https://example.com/> qui a fait l'appel suivant :

```
navigator.registerProtocolHandler('web+soup', 'soup?url=%s')
```

...puis, bien plus tard, en visitant <https://www.example.net/>, cliqué sur un lien tel que :

```
<a href="web+soup:chicken-kiwi">Download our Chicken Kiwi  
soup!</a>
```

... alors l'UA peut accéder à l'URL suivante :

```
https://example.com/soup?url=web+soup:chicken-k%C3%AFwi
```

Ce site pourrait alors faire ce qu'il fait avec la soupe (la synthétiser et l'expédier à l'utilisateur, ou quoi que ce soit).

Cela ne définit pas quand le gestionnaire est utilisé. Dans une certaine mesure, le [modèle de traitement pour naviguer dans les documents](#) définit certains cas où cela est pertinent, mais en général, les agents utilisateurs

peuvent utiliser ces informations partout où ils envisageraient autrement de transmettre des schémas à des plugins natifs ou à des applications d'assistance.

Les étapes de la méthode sont : `unregisterProtocolHandler(scheme, url)`

1. Soit ( *normalizedScheme* , *normalizedURLString* ) le résultat de l'exécution [des paramètres de normalisation du gestionnaire de protocole](#) avec le schéma , l' url et [cet objet de paramètres](#) pertinent .
  2. [En parallèle](#) : désinscrire le gestionnaire décrit par *normalizedScheme* et *normalizedURLString* .
- 

Pour **normaliser les paramètres du gestionnaire de protocole** , en fonction d'un schéma de chaîne , d'une URL de chaîne et d'un environnement [d'objet de paramètres d'environnement](#) , exécutez ces étapes :

1. Définissez *schéma* sur *schéma* , [converti en minuscules ASCII](#) .
2. Si le schéma n'est ni un [schéma de la liste sécurisée](#) ni une chaîne commençant par " web+" suivi d'un ou plusieurs [caractères alpha inférieurs ASCII](#) , lancez alors un "[SecurityError](#)" [DOMException](#) .

*Cela signifie que l'inclusion de deux-points dans le schéma (comme dans "mailto:") lancera.*

Les schémas suivants sont les **schémas safelistés** :

- o bitcoin
- o ftp
- o ftps
- o geo
- o im
- o irc
- o ircs
- o magnet
- o mailto
- o matrix
- o mms
- o news
- o nntp
- o openpgp4fpr
- o sftp
- o sip
- o sms
- o smsto
- o ssh
- o tel
- o urn

- o webcal
- o wtai
- o xmpp

*Cette liste peut être modifiée. S'il y a des schémas qui doivent être ajoutés, veuillez envoyer vos commentaires.*

3. Si l'url ne contient pas "%s", lancez un `"SyntaxError" DOMException`.
4. [Analyser](#) l'URL par rapport à l'environnement.
5. Si cela échoue, lancez un `"SyntaxError" DOMException`.

*C'est forcément le cas si l'%s espace réservé se trouve dans l'hôte ou le port de l'URL.*

6. Si le [schéma](#) de l'enregistrement d' [URL résultant](#) n'est pas un [schéma HTTP\(S\)](#) ou si l' [origine](#) de l'enregistrement d' [URL résultant](#) n'est pas [la même origine](#) que l'[origine](#) de l' [environnement](#), lancez un `"SecurityError DOMException"`.
7. [Assert](#) : le résultat de [L'url est-elle potentiellement digne de confiance ?](#) étant donné que [l'enregistrement d'URL résultant](#) est "Potentially Trustworthy".

*Comme [normalize protocol handler parameters](#) est exécuté dans un [contexte sécurisé](#), cela est impliqué par la [même](#) condition d'origine.*

8. Retourne ( [schéma](#), [chaîne d'URL résultante](#) ).

*La [chaîne d'URL résultante](#) ne sera par définition pas une [chaîne d'URL valide](#) car elle inclut la chaîne "%s" qui n'est pas un composant valide dans une URL.*

#### 8.9.1.4.1 Sécurité et confidentialité

Les gestionnaires de schémas personnalisés peuvent introduire un certain nombre de problèmes, en particulier des problèmes de confidentialité.

**Détournement de toute utilisation du Web.** Les agents utilisateurs ne doivent pas permettre aux schémas qui sont essentiels à son fonctionnement normal, tels qu'un [schéma HTTP\(S\)](#), d'être redirigés via des sites tiers. Cela permettrait de suivre de manière triviale les activités d'un utilisateur et de collecter des informations sur l'utilisateur, même dans des connexions sécurisées.

**Détournement des valeurs par défaut.** Les agents utilisateurs sont fortement invités à ne pas modifier automatiquement les valeurs par défaut, car cela pourrait conduire l'utilisateur à envoyer des données à des hôtes distants auxquels l'utilisateur ne s'attend pas. Les nouveaux gestionnaires qui s'enregistrent ne doivent jamais entraîner automatiquement l'utilisation de ces sites.

**Spam d'inscription.** Les agents utilisateurs doivent envisager la possibilité qu'un site tente d'enregistrer un grand nombre de gestionnaires, éventuellement à partir de plusieurs domaines (par exemple, en redirigeant vers une série de pages chacune sur un domaine différent, et chacun enregistrant un gestionnaire pour - des pratiques analogues abusant d' `web+spam`: autres les fonctionnalités des navigateurs Web sont utilisées par les sites Web pornographiques depuis de nombreuses années). Les agents utilisateurs doivent gérer avec élégance de telles tentatives hostiles, en protégeant l'utilisateur.

**Métadonnées du gestionnaire hostile.** Les agents utilisateurs doivent se protéger contre les attaques typiques contre les chaînes intégrées dans leur interface, par exemple en s'assurant que les caractères de balisage ou d'échappement dans ces chaînes ne sont pas exécutés, que les octets nuls sont correctement gérés, que les chaînes trop longues ne provoquent pas de plantages ou de dépassements de mémoire tampon, et ainsi de suite.

**Fuite de données privées.** Les auteurs de pages Web peuvent faire référence à un gestionnaire de schéma personnalisé à l'aide de données d'URL considérées comme privées. Ils peuvent le faire en s'attendant à ce que le gestionnaire choisi par l'utilisateur pointe vers une page à l'intérieur de l'organisation, garantissant ainsi que les données sensibles ne seront pas exposées à des tiers. Cependant, un utilisateur peut avoir enregistré un gestionnaire pointant vers un site externe, entraînant une fuite de données vers ce tiers. Les implémenteurs pourraient envisager d'autoriser les administrateurs à désactiver les gestionnaires personnalisés sur certains sous-domaines, types de contenu ou schémas.

**Interférence d'interface.** Les agents utilisateurs doivent être prêts à gérer des arguments intentionnellement longs pour les méthodes. Par exemple, si l'interface utilisateur exposée se compose d'un bouton "accepter" et d'un bouton "refuser", avec la liaison "accepter" contenant le nom du gestionnaire, il est important qu'un nom long ne provoque pas l'activation du bouton "refuser". poussé hors de l'écran.

#### 8.9.1.4.2 Automatisation de l'agent utilisateur

Chacun [Document](#) a un `registerProtocolHandler()` **mode d'automatisation** . Sa valeur par défaut est " [none](#)", mais il peut également s'agir de " [auto-accept](#)" ou de " [auto-reject](#)".

Aux fins de l'automatisation de l'agent utilisateur et des tests de sites Web, cette norme définit [la commande d'extension](#) **Set RPH Registration Mode** WebDriver . Il demande à l'agent utilisateur de placer a dans un mode où il simulera automatiquement un utilisateur acceptant ou rejetant une boîte de dialogue d'invite de confirmation d'enregistrement. [Document](#)

Méthode HTTP	Modèle d'URI
<code>POST</code>	<code>/session/{session id}/custom-handlers/set-mode</code>

Les [étapes de fin à distance](#) sont :

1. Si *parameters* n'est pas un objet JSON, renvoie une [erreur WebDriver](#) avec [l'argument non valide du code d'erreur WebDriver](#) .
2. Soit *mode* le résultat de [l'obtention d'une propriété](#) nommée " *mode* " à partir de *parameters* .
3. Si *mode* n'est pas " [auto-accept](#) ", " [auto-reject](#) " ou " [none](#) ", renvoie une [erreur WebDriver](#) avec [un argument invalide de code d'erreur WebDriver](#) .
4. Soit *document* le [document actif](#) du [contexte de navigation actuel](#) .
5. Définissez [le mode d'automatisation](#) du *document* sur *mode* [.registerProtocolHandler\(\)](#)
6. Renvoie [le succès](#) avec des données nulles.

#### 8.9.1.5 Cookies

```
interface mixin NavigatorCookies {
```

```
    readonly attribute boolean cookieEnabled;
```

```
};
```

**[window.navigator.cookieEnabled](#)**

✓

Renvoie false si la définition d'un cookie sera ignorée et true dans le cas contraire.

L' [cookieEnabled](#) attribut doit renvoyer true si l'agent utilisateur tente de gérer les cookies conformément au *mécanisme de gestion d'état HTTP* et false s'il ignore les demandes de modification des cookies. [\[BISCUITS\]](#)

#### 8.9.1.6 Prise en charge de l'affichage PDF

**[window.navigator.pdfViewerEnabled](#)**

Renvoie true si l'agent utilisateur prend en charge l'affichage en ligne des fichiers PDF lors de [la navigation](#) vers ceux-ci, ou false sinon. Dans ce dernier cas, les fichiers PDF seront gérés par [un logiciel externe](#) .

```
interface mixin NavigatorPlugins {
```

```
[SameObject] readonly attribute PluginArray plugins;
```

```
[SameObject] readonly attribute MimeTypeArray mimeTypes;
```

```
boolean javaEnabled();
```

```
readonly attribute boolean pdfViewerEnabled;
```

```
};
```

```
[Exposed=Window,
```

```
LegacyUnenumerableNamedProperties]
```

```
interface PluginArray {
```

```
    undefined refresh();
```

```
    readonly attribute unsigned long length;
```

```
    getter Plugin? item(unsigned long index);
```

```
    getter Plugin? namedItem(DOMString name);
```

```
};
```

```
[Exposed=Window,
```

```
LegacyUnenumerableNamedProperties]
```

```
interface MimeTypeArray {
```

```
    readonly attribute unsigned long length;
```

```
    getter MimeType? item(unsigned long index);
```

```
    getter MimeType? namedItem(DOMString name);
```

```
};
```

```
[Exposed=Window,
```

```
LegacyUnenumerableNamedProperties]
```

```
interface Plugin {
```

```
    readonly attribute DOMString name;
```

```
    readonly attribute DOMString description;
```

```
    readonly attribute DOMString filename;
```

```
    readonly attribute unsigned long length;
```

```
    getter MimeType? item(unsigned long index);
```

```
    getter MimeType? namedItem(DOMString name);
```

```
};
```

```
[Exposed=Window]
```

```
interface MimeType {
```

```
    readonly attribute DOMString type;
```

```
    readonly attribute DOMString description;
```

```
    readonly attribute DOMString suffixes;
```

```
    readonly attribute Plugin enabledPlugin;
```

```
};
```

Bien que de nos jours, la détection de la prise en charge de la visionneuse PDF puisse être effectuée via [navigator.pdfViewerEnabled](#), pour des raisons historiques, il existe un certain nombre d'interfaces complexes et entrelacées qui offrent la même capacité, sur laquelle repose le code hérité. Cette section spécifie à la fois la variante moderne simple et la variante historique compliquée.

Chaque agent utilisateur dispose d'un booléen **pris en charge par le visualiseur PDF**, dont la valeur est [définie par l'implémentation](#) (et peut varier en fonction des préférences de l'utilisateur).

*Cette valeur impacte également le modèle de traitement [de la navigation](#).*

---

Chaque [Window](#) objet a une liste **d'objets de plug-in de visionneuse PDF** . Si le [visualiseur PDF de l'agent utilisateur pris en charge](#) est faux, il s'agit de la liste vide. Sinon, il s'agit d'une liste contenant cinq [Plugin](#) objets, dont [les noms](#) sont respectivement :

1. " PDF Viewer"
2. " Chrome PDF Viewer"
3. " Chromium PDF Viewer"
4. " Microsoft Edge PDF Viewer"
5. " WebKit built-in PDF"

Les valeurs de la liste ci-dessus forment la liste **des noms des plug-ins de la visionneuse PDF** .

*Ces noms ont été choisis sur la base de preuves de ce que les sites Web recherchent historiquement, et donc de ce que les agents utilisateurs doivent exposer afin de maintenir la compatibilité avec le contenu existant. Ils sont classés par ordre alphabétique. Le PDF Viewer nom " " a ensuite été inséré en 0ème position afin que le [enabledPlugin](#) getter puisse pointer vers un nom de plugin générique.*

Chaque [Window](#) objet a une liste **d'objets de type mime de visionneuse PDF** . Si le [visualiseur PDF de l'agent utilisateur pris en charge](#) est faux, il s'agit de la liste vide. Sinon, il s'agit d'une liste contenant deux [MimeType](#) objets, dont [les types](#) sont respectivement :

1. " application/pdf"
2. " text/pdf"

Les valeurs de la liste ci-dessus forment la liste **des types mime de la visionneuse PDF** .

---

Chaque [NavigatorPlugins](#) objet a un **tableau plugins** , qui est un nouveau [PluginArray](#), et un **tableau de types mime** , qui est un nouveau [MimeTypeArray](#).

Les étapes `getter` [NavigatorPlugins](#) du mixin **plugins** consistent à renvoyer [le tableau de plugins](#) de [this](#) .

Les étapes de `getter` [NavigatorPlugins](#) du mixin **mimeTypes** consistent à renvoyer [le tableau de types mime](#) de [this](#) .



Les étapes de la méthode `NavigatorPlugins` mixin `javaEnabled()` consistent à retourner false.

## MDN

Les étapes du getter `NavigatorPlugins` du mixin `pdfViewerEnabled` consistent à renvoyer la [visionneuse PDF de l'agent utilisateur prise en charge](#).

---

L'interface `PluginArray` [prend en charge les propriétés nommées](#). Si le [visualiseur PDF de l'agent utilisateur pris en charge](#) est vrai, il s'agit [des noms de plug-in du visualiseur PDF](#). Sinon, ils sont la liste vide.

Les étapes `PluginArray` de la méthode de l'interface sont :`namedItem(name)`

1. Pour chaque `Plugin` *plug-in* des [objets de plug-in de visionneuse PDF](#) de [cet objet global pertinent](#) : si [le nom](#) du *plug-in* est *nom*, alors renvoie *plug-in*.
2. Renvoie nul.

L'interface `PluginArray` [prend en charge les propriétés indexées](#). Les [indices de propriétés pris en charge](#) sont les [indices](#) des [objets du plug-in de visionneuse PDF](#) de [cet objet global pertinent](#).

Les étapes `PluginArray` de la méthode de l'interface sont :`item(index)`

1. Supposons que *les plugins* soient les objets du [plugin de visionneuse PDF](#) de [cet objet global pertinent](#).
2. Si *index* < [taille](#) des *plugins*, alors retourne *plugins* [ *index* ].
3. Renvoie nul.

Les étapes du getter `PluginArray` de l'interface `length` consistent à renvoyer la [taille](#) des [objets du plug-in de visionneuse PDF](#) de [cet objet global pertinent](#).

Les étapes de la méthode `PluginArray` de l'interface `refresh()` consistent à ne rien faire.

---

L' `MimeTypeArray` interface [prend en charge les propriétés nommées](#) . Si le [visualiseur PDF de l'agent utilisateur pris en charge](#) est vrai, il s'agit alors des [types mime du visualiseur PDF](#) . Sinon, ils sont la liste vide.

Les étapes `MimeTypeArray` de la méthode de l'interface sont : `namedItem(name)`

1. Pour chaque `MimeType`  *mimeType*  des objets de type mime de [la visionneuse PDF de cet objet global pertinent](#) : si [le type](#) de  *mimeType*  est  *name*  , alors renvoie  *mimeType*  .
2. Renvoie nul.

L' `MimeTypeArray` interface [prend en charge les propriétés indexées](#) . Les [index de propriété pris en charge](#) sont les [index](#) des [objets de type mime](#) de la visionneuse PDF de [cet objet global pertinent](#) .

Les étapes `MimeTypeArray` de la méthode de l'interface sont : `item(index)`

1. Soit  *mimeTypees*  les objets de type MIME du [visualiseur PDF](#) de [cet objet global pertinent](#) .
2. Si  *index*  < [taille](#) de  *mimeTypees*  , alors retourne  *mimeTypees [ index ]* .
3. Renvoie nul.

Les étapes getter `MimeTypeArray` de l'interface `length` consistent à renvoyer [la taille](#) des objets de type mime de [la visionneuse PDF](#) de [cet objet global pertinent](#) .

---

Chaque `Plugin` objet a un **nom** , qui est défini lors de la création de l'objet.

Les étapes du getter `Plugin` de l'interface `name` consistent à renvoyer [this](#) 's [name](#) .

Les étapes du getter `Plugin` de l'interface `description` consistent à renvoyer " Portable Document Format ".

Les étapes du getter `Plugin` de l'interface `filename` consistent à renvoyer " internal-pdf-viewer ".

L' `Plugin` interface [prend en charge les propriétés nommées](#) . Si le [visualiseur PDF de l'agent utilisateur pris en charge](#) est vrai, il s'agit alors des [types mime du visualiseur PDF](#) . Sinon, ils sont la liste vide.

Les étapes `Plugin` de la méthode de l'interface sont : `namedItem(name)`

1. Pour chaque `MimeType`  *mimeType*  des objets de type mime de [la visionneuse PDF de cet](#) objet global [pertinent](#) : si [le type](#) de  *mimeType*  est  *name* , alors renvoie  *mimeType* .
2. Renvoie nul.

L' `Plugin` interface [prend en charge les propriétés indexées](#). Les [index de propriété pris en charge](#) sont les [index](#) des [objets de type mime](#) de la visionneuse PDF de [cet objet global pertinent](#).

Les étapes `Plugin` de la méthode de l'interface sont : `item(index)`

1. Soit  *mimeType*  les objets de type MIME du [visualiseur PDF](#) de [cet objet global pertinent](#).
2. Si  *index*  < [taille](#) de  *mimeType* , alors retourne  *mimeType [ index ]*.
3. Renvoie nul.

Les étapes `getter Plugin` de l'interface `length` consistent à renvoyer [la taille](#) des objets de type mime de [la visionneuse PDF](#) de [cet objet global pertinent](#).

---

Chaque `MimeType` objet a un **type**, qui est défini lors de la création de l'objet.

Les étapes du `getter MimeType` de l'interface `type` consistent à renvoyer [this](#) 's [type](#).

Les étapes du `getter MimeType` de l'interface `description` consistent à renvoyer " Portable Document Format".

Les étapes du `getter MimeType` de l'interface `suffixes` consistent à renvoyer " pdf".

Les étapes du `getter MimeType` de l'interface `enabledPlugin` consistent à renvoyer [les objets du plug-in de visionneuse PDF](#) de [cet](#) objet [global pertinent](#) [0] (c'est-à-dire, le générique " ").PDF Viewer

1.

1. [8.10 Images](#)
2. [8.11 Images animées](#)

## 8.10 Images

✓ MDN

```
[Exposed=(Window,Worker), Serializable, Transferable]
```

```
interface ImageBitmap {
```

```
  readonly attribute unsigned long width;
```

```
  readonly attribute unsigned long height;
```

```
  undefined close();
```

```
};
```

```
typedef (CanvasImageSource or
```

```
  Blob or
```

```
  ImageData) ImageBitmapSource;
```

```
enum ImageOrientation { "from-image", "flipY" };
```

```
enum PremultiplyAlpha { "none", "premultiply", "default" };
```

```
enum ColorSpaceConversion { "none", "default" };
```

```
enum ResizeQuality { "pixelated", "low", "medium", "high" };
```

```
dictionary ImageBitmapOptions {
```

```
  ImageOrientation imageOrientation = "from-image";
```

```
PremultiplyAlpha premultiplyAlpha = "default";
```

```
ColorSpaceConversion colorSpaceConversion = "default";
```

```
[EnforceRange] unsigned long resizeMode;
```

```
[EnforceRange] unsigned long resizeMode;
```

```
ResizeQuality resizeMode = "low";
```

```
};
```

Un [ImageBitmap](#) objet représente une image bitmap qui peut être peinte sur un canevas sans latence excessive.

*Le jugement exact de ce qui est une latence excessive est laissé à l'implémenteur, mais en général, si l'utilisation du bitmap nécessite des E/S réseau, ou même des E/S disque locales, alors la latence est probablement excessive ; alors que s'il ne nécessite qu'une lecture bloquante à partir d'un GPU ou d'une RAM système, la latence est probablement acceptable.*

```
promise = self.createImageBitmap(image [, options ])
```



```
promise = self.createImageBitmap(image, sx, sy, sw, sh [, options ])
```

Prend *image* , qui peut être un [img](#) élément, un élément [SVG imagevideo](#) , un élément, un [canvas](#) élément, un [Blob](#) objet, un [ImageData](#) objet ou un autre [ImageBitmap](#) objet, et renvoie une promesse qui est résolue lorsqu'une nouvelle [ImageBitmap](#) est créée.

Si aucun [ImageBitmap](#) objet ne peut être construit, par exemple parce que les données *d'image* fournies ne sont pas réellement une image, alors la promesse est rejetée à la place.

Si les arguments *sx* , *sy* , *sw* et *sh* sont fournis, l'image source est recadrée aux pixels donnés, tous les pixels manquants dans l'original étant remplacés par [du noir transparent](#) . Ces coordonnées sont dans l'espace de coordonnées en pixels de l'image source, *et non* en [pixels CSS](#) .

Si *options* est fourni, les [ImageBitmap](#) données bitmap de l'objet sont modifiées selon *options* . Par exemple, si l' [premultiplyAlpha](#) option est définie sur "[premultiply](#)", les canaux de couleur des [données bitmap](#) sont [prémultipliés par leur canal alpha](#) .

Rejette la promesse avec un "[InvalidStateError](#)" [DOMException](#) si l'image source n'est pas dans un état valide (par exemple, un [img](#) élément qui n'a pas été chargé avec succès, un [ImageBitmap](#) objet dont la valeur d'emplacement interne [\[\[Detached\]\]ImageData](#) est true, un objet dont [data](#) la valeur d'attribut est [\[\[ViewedArrayBuffer\]\]](#) emplacement interne est détaché ou [Blob](#) dont les données ne peuvent pas être interprétées comme une image bitmap).

Rejette la promesse avec un "[SecurityError](#)" [DOMException](#) si le script n'est pas autorisé à accéder aux données d'image de l'image source (par exemple, a [video](#) qui est [CORS-cross-origin](#) , ou a [canvas](#) étant dessiné par un script dans un worker d'une autre [origine](#) ).

`imageBitmap.close()`



Libère [les données bitmap](#) sous-jacentes d' *imageBitmap* .

`imageBitmap.width`



Renvoie la [largeur intrinsèque](#) de l'image, en [pixels CSS](#) .

`imageBitmap.height`



Renvoie la [hauteur intrinsèque](#) de l'image, en [pixels CSS](#) .

Un [ImageBitmap](#) objet dont la valeur de slot interne [\[\[Detached\]\]](#) est false a toujours **des données bitmap** associées , avec une largeur et une hauteur. Cependant, il est possible que ces données soient corrompues. Si [ImageBitmap](#) les données multimédia d'un objet peuvent être décodées sans erreur, on dit qu'elles sont **entièrement décodables** .

Le bitmap d'un [ImageBitmap](#) objet a un indicateur [origin-clean](#) , qui indique si le bitmap est entaché par du contenu d'une [origine](#) différente . L'indicateur est initialement défini sur true et peut être modifié sur false par les étapes de [createImageBitmap\(\)](#) .

---

[ImageBitmap](#) les objets sont [des objets sérialisables](#) et [des objets transférables](#) .

Leurs [étapes de sérialisation](#) , *valorisées* et *sérialisées* , sont :

1. Si l'indicateur [origin-clean](#) de *value* n'est pas défini, lancez un "[DataCloneError](#)" [DOMException](#)
2. Définissez *serialized* .[[[BitmapData](#)]] sur une copie des [données bitmap](#) de la *valeur* .

Leurs [étapes de désérialisation](#) , étant donné *serialized* , *value* et *targetRealm* , sont :

1. Définissez [les données bitmap](#) de la *valeur* sur *sérialisé* .[[[BitmapData](#)]].

Leurs [étapes de transfert](#) , compte tenu de *value* et *dataHolder* , sont :

1. Si l'indicateur [origin-clean](#) de *value* n'est pas défini, lancez un `"` . `DataCloneError DOMException`
2. Définissez *dataHolder* .[[*BitmapData*]] sur [les données bitmap](#) de la *valeur* .
3. [Données bitmap](#) de la *valeur* non définie .

Leurs [étapes de réception de transfert](#) , étant donné *dataHolder* et *value* , sont :

1. Définissez [les données bitmap](#) de la *valeur* sur *dataHolder* .[[*BitmapData*]].

---

Les méthodes et , lorsqu'elles sont appelées, doivent exécuter ces

étapes :`createImageBitmap(image, options)``createImageBitmap(image sx, sy, sw, sh, options)`

1. Si *sw* ou *sh* est donné et vaut 0, alors renvoie [une promesse rejetée avec](#) a `RangeError`.
2. Si l'une ou l'autre des *options*`resizeWidth` ou des *options*`resizeHeight` est présente et vaut 0, alors renvoie [une promesse rejetée avec](#) un `" InvalidStateError" DOMException` .
3. [Vérifiez l'utilisabilité de l' argument image](#) . Si cela lève une exception ou retourne *bad* , alors retournez [une promesse rejetée avec](#) un `" InvalidStateError" DOMException` .
4. Soit *p* une nouvelle promesse.
5. Soit *imageBitmap* un nouvel `ImageBitmap` objet.
6. Activer l'image :

`img`

[SVG image](#)

1. Si les données multimédias de l' *image* n'ont pas [de dimensions intrinsèques](#) (par exemple, c'est un graphique vectoriel sans taille de contenu spécifiée) et que les *options*`resizeWidth` ou les *options* `ne``resizeHeight` sont pas présentes, alors [une promesse est rejetée avec](#) un `" InvalidStateError" DOMException` .
2. Si les données multimédias de l'*image* n'ont pas [de dimensions intrinsèques](#) (par exemple, s'il s'agit d'un graphique vectoriel sans taille

de contenu spécifiée), elles doivent être rendues dans un bitmap de la taille spécifiée par les options `resizeWidth` et `resizeHeight` .

3. Définissez [les données bitmap](#) de `imageBitmap` sur une copie des données multimédias de `l'image` , [recadrées au rectangle source avec le formatage](#) . S'il s'agit d'une image animée, [les données bitmap](#) de `imageBitmap` doivent uniquement provenir de l'image par défaut de l'animation (celle que le format définit doit être utilisée lorsque l'animation n'est pas prise en charge ou est désactivée), ou, s'il n'y a pas telle image, la première image de l'animation.
4. Si `image` [n'est pas origin-clean](#) , définissez l' indicateur [origin-clean du bitmap de imageBitmap](#) sur `false`.
5. Exécutez cette étape [en parallèle](#) :
  1. Résolvez `p` avec `imageBitmap` .

#### video

6. Si l'attribut de l' `image` `networkState` est `NETWORK_EMPTY`, alors retourne [une promesse rejetée avec](#) un `"InvalidStateError"` `DOMException` .
7. Définissez [les données bitmap](#) de `imageBitmap` sur une copie de l'image à la [position de lecture actuelle](#) , à [la largeur intrinsèque](#) et [à la hauteur intrinsèque](#) de [la ressource multimédia](#) (c'est-à-dire après l'application de toute correction de rapport d'aspect), [recadrée au rectangle source avec mise en forme](#) .
8. Si `image` [n'est pas origin-clean](#) , définissez l' indicateur [origin-clean du bitmap de imageBitmap](#) sur `false`.
9. Exécutez cette étape [en parallèle](#) :
  1. Résolvez `p` avec `imageBitmap` .

#### canvas

10. Définissez [les données bitmap](#) de `imageBitmap` sur une copie des [données bitmap](#) de `l'image` , [rognées sur le rectangle source avec formatage](#) .
11. Définissez le drapeau [origin-clean](#) du bitmap de `imageBitmap` sur la même valeur que le drapeau [origin-clean du bitmap de image](#) .
12. Exécutez cette étape [en parallèle](#) :
  1. Résolvez `p` avec `imageBitmap` .

#### Blob

Exécutez ces étapes [en parallèle](#) :



13. Soit *imageData* le résultat de la lecture des données de l' *image* . Si une [erreur survient lors de la lecture de l'objet](#) , alors rejetez *p* avec un ["InvalidStateError"](#) [DOMException](#) et abandonnez ces étapes.
14. Appliquez les [règles de reniflage d'image](#) pour déterminer le format de fichier de *imageData* , avec le type MIME d' *image* (tel qu'indiqué par l' attribut *image*[type](#) ) donnant le type officiel.
15. Si *imageData* n'est pas dans un format de fichier image pris en charge (par exemple, ce n'est pas du tout une image), ou si *imageData* est corrompu d'une manière fatale telle que les dimensions de l'image ne peuvent pas être obtenues (par exemple, un graphique vectoriel sans taille intrinsèque), puis rejetez *p* avec un ["InvalidStateError"](#) [DOMException](#) et annulez ces étapes.
16. Définissez [les données bitmap](#) d' *imageBitmap* sur *imageData* , [rognées sur le rectangle source avec mise en forme](#) . S'il s'agit d'une image animée, [les données bitmap](#) de *imageBitmap* doivent uniquement provenir de l'image par défaut de l'animation (celle que le format définit doit être utilisée lorsque l'animation n'est pas prise en charge ou est désactivée), ou, s'il n'y a pas telle image, la première image de l'animation.
17. Résolvez *p* avec *imageBitmap* .

#### ImageData

18. Soit *buffer* l' emplacement interne `[[ViewedArrayBuffer]]` de la valeur d'attribut de l'*image*[data](#) .
19. Si [IsDetachedBuffer](#) ( *buffer* ) est vrai, renvoie [une promesse rejetée avec](#) un ["InvalidStateError"](#) [DOMException](#) .
20. Définissez [les données bitmap](#) de *imageBitmap* sur les données d'image de l' *image* , [recadrées au rectangle source avec le formatage](#) .
21. Exécutez cette étape [en parallèle](#) :
  1. Résolvez *p* avec *imageBitmap* .

#### ImageBitmap

22. Définissez [les données bitmap](#) de *imageBitmap* sur une copie des [données bitmap](#) de l'*image* , [rognées sur le rectangle source avec formatage](#) .
23. Définissez le drapeau [origin-clean du bitmap de](#) *imageBitmap* sur la même valeur que le drapeau [origin-clean du bitmap de](#) *image* .
24. Exécutez cette étape [en parallèle](#) :
  1. Résolvez *p* avec *imageBitmap* .

## VideoFrame

25. Définissez les données bitmap de *imageBitmap* sur une copie des données de pixels visibles de l'image , recadrées au rectangle source avec formatage .

26. Exécutez cette étape en parallèle :

1. Résolvez *p* avec *imageBitmap* .

7. Retour *p* .

Lorsque les étapes ci-dessus nécessitent que l'agent utilisateur **recadre les données bitmap dans le rectangle source avec formatage** , l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit l'entrée les données bitmap en cours de transformation.
2. Si *sx* , *sy* , *sw* et *sh* sont spécifiés, soit *sourceRectangle* un rectangle dont les coins sont les quatre points ( *sx* , *sy* ), ( *sx* + *sw* , *sy* ), ( *sx* + *sw* , *sy* + *sh* ), ( *sx* , *sy* + *ch* ). Sinon, laissez *sourceRectangle* être un rectangle dont les coins sont les quatre points (0,0), (width of *input* , 0), (width of *input* , height of *input*), (0, hauteur de l'entrée ).

*Si sw ou sh sont négatifs, alors le coin supérieur gauche de ce rectangle sera à gauche ou au-dessus du point ( sx , sy ).*

3. Laissez *outputWidth* être déterminé comme suit :

**Si le resizeWidth membre d' options est spécifié**

la valeur du resizeWidth membre des *options*

**Si le resizeWidth membre d' options n'est pas spécifié, mais que le resizeHeight membre est spécifié**

la largeur de *sourceRectangle* , multipliée par la valeur du resizeHeight membre de *options* , divisée par la hauteur de *sourceRectangle* , arrondie à l'entier le plus proche

**Si ni resizeWidth ni ne resizeHeight sont spécifiés**

la largeur de *sourceRectangle*

4. Laissez *outputHeight* être déterminé comme suit :

**Si le resizeHeight membre d' options est spécifié**

la valeur du resizeHeight membre des *options*

**Si le resizeHeight membre d' options n'est pas spécifié, mais que le resizeWidth membre est spécifié**

la hauteur de *sourceRectangle* , multipliée par la valeur du resizeWidth membre de *options* , divisée par la largeur de *sourceRectangle* , arrondie à l'entier le plus proche

Si ni `resizeWidth` ni `resizeHeight` sont spécifiés

la hauteur de `sourceRectangle`

5. Placez l'entrée sur un plan de grille [noir transparent](#) infini , positionné de sorte que son coin supérieur gauche soit à l'origine du plan, avec la coordonnée x augmentant vers la droite et la coordonnée y augmentant vers le bas, et avec chaque pixel dans l' *entrée* données d'image occupant une cellule sur la grille du plan.
6. Soit *output* le rectangle sur le plan désigné par `sourceRectangle` .
7. Mettez à l'échelle la sortie à la taille spécifiée par `outputWidth` et `outputHeight` . L'agent utilisateur devrait utiliser la valeur de l' `resizeQuality` option pour guider le choix de l'algorithme de mise à l'échelle.
8. Si la valeur du `imageOrientation` membre des *options* est " `flipY` ", la sortie doit être inversée verticalement, sans tenir compte des métadonnées d'orientation de l'image de la source (telles que les métadonnées EXIF), le cas échéant. [\[EXIF\]](#)

*Si la valeur est " `from-image` ", aucune étape supplémentaire n'est nécessaire. Il y avait une `none` valeur d'énumération " ". Il a été renommé " `from-image` ". À l'avenir, " `none` " sera rajouté avec une signification différente.*

9. Si *image* est un `img` élément ou un `Blob` objet, laissez *val* être la valeur du `colorSpaceConversion` membre de *options* , puis exécutez ces sous-étapes :
  1. Si *val* est " `default` ", le comportement de conversion de l'espace colorimétrique est spécifique à l'implémentation et doit être choisi en fonction de l'espace colorimétrique par défaut utilisé par l'implémentation pour dessiner des images sur le canevas.
  2. Si *val* est " `none` ", la sortie doit être décodée sans effectuer de conversion d'espace colorimétrique. Cela signifie que l'algorithme de décodage d'image doit ignorer les métadonnées du profil de couleur intégrées dans les données source ainsi que le profil de couleur du dispositif d'affichage.
10. Soit *val* la valeur de `premultiplyAlpha` member of *options* , puis exécutez ces sous-étapes :
  1. Si *val* est " `default` ", le comportement de prémultiplication alpha est spécifique à l'implémentation et doit être choisi en fonction de l'implémentation jugée optimale pour dessiner des images sur le canevas.
  2. Si *val* est " `premultiply` ", la sortie qui n'est pas prémultipliée par alpha doit avoir ses composants de couleur [multipliés par alpha](#) et qui est prémultipliée par alpha doit rester intacte.

3. Si `val` est "`none`", la *sortie* qui n'est pas prémultipliée par alpha doit rester intacte et qui est prémultipliée par alpha doit avoir ses composants de couleur [divisés par alpha](#) .

11. *Sortie* de retour .

Les `close()` étapes de la méthode sont :

1. Définissez la valeur de `cet` emplacement interne [\[\[Detached\]\]](#) sur `true`.
2. Désélectionnez `ces` données `bitmap` .

Les `width` étapes du getter sont :

1. Si la valeur de `cet` emplacement interne [\[\[Detached\]\]](#) est vraie, alors renvoyez [0](#).
2. Renvoie la largeur de [this , en pixels CSS](#) .

Les `height` étapes du getter sont :

1. Si la valeur de `cet` emplacement interne [\[\[Detached\]\]](#) est vraie, alors renvoyez [0](#).
2. Renvoie la hauteur de [this , en pixels CSS](#) .

L' `ResizeQuality` énumération est utilisée pour exprimer une préférence pour la qualité d'interpolation à utiliser lors de la mise à l'échelle des images.

La `pixelated` valeur " " indique une préférence de mise à l'échelle de l'image qui maximise l'apparence. Les algorithmes de mise à l'échelle qui "lissent" les couleurs sont acceptables, comme l'interpolation bilinéaire.

La `low` valeur " " indique une préférence pour un faible niveau de qualité d'interpolation d'image. Une interpolation d'image de faible qualité peut être plus efficace en termes de calcul que des paramètres plus élevés.

La `medium` valeur " " indique une préférence pour un niveau moyen de qualité d'interpolation d'image.

La `high` valeur " " indique une préférence pour un niveau élevé de qualité d'interpolation d'image. Une interpolation d'image de haute qualité peut être plus coûteuse en calcul que des paramètres inférieurs.

*La mise à l'échelle bilinéaire est un exemple d'algorithme de lissage d'image relativement rapide et de moindre qualité. La mise à l'échelle bicubique ou Lanczos sont des exemples d'algorithmes de mise à l'échelle d'image qui produisent une sortie de meilleure qualité. Cette spécification n'impose pas l'utilisation d'algorithmes d'interpolation spécifiques à moins que la valeur ne soit " [pixélisée](#) ".*

À l'aide de cette API, une feuille de sprite peut être prédécoupée et préparée :

```
var sprites = {};  
function loadMySprites() {  
  var image = new Image();  
  image.src = 'mysprites.png';  
  var resolver;  
  var promise = new Promise(function (arg) { resolver = arg });  
  image.onload = function () {  
    resolver(Promise.all([  
      createImageBitmap(image, 0, 0, 40, 40).then(function  
(image) { sprites.person = image }),  
      createImageBitmap(image, 40, 0, 40, 40).then(function  
(image) { sprites.grass = image }),  
      createImageBitmap(image, 80, 0, 40, 40).then(function  
(image) { sprites.tree = image }),  
      createImageBitmap(image, 0, 40, 40, 40).then(function  
(image) { sprites.hut = image }),  
      createImageBitmap(image, 40, 40, 40, 40).then(function  
(image) { sprites.apple = image }),  
      createImageBitmap(image, 80, 40, 40, 40).then(function  
(image) { sprites.snake = image })  
    ]));  
  };  
  return promise;  
}  
  
function runDemo() {  
  var canvas = document.querySelector('canvas#demo');  
  var context = canvas.getContext('2d');  
  context.drawImage(sprites.tree, 30, 10);  
  context.drawImage(sprites.snake, 70, 10);  
}  
  
loadMySprites().then(runDemo);
```

## 8.11 Images animées

Certains objets incluent l' [AnimationFrameProvider](#) interface mixin.

```
callback FrameRequestCallback = undefined (DOMHighResTimeStamp  
time);
```

```
interface mixin AnimationFrameProvider {
```

```
    unsigned long requestAnimationFrame (FrameRequestCallback  
callback);
```

```
    undefined cancelAnimationFrame (unsigned long handle);
```

```
};
```

```
Window includes AnimationFrameProvider;
```

```
DedicatedWorkerGlobalScope includes AnimationFrameProvider;
```

Chaque [AnimationFrameProvider](#) objet possède également un **objet cible** qui stocke l'état interne du fournisseur. Il est défini comme suit :

Si le [AnimationFrameProvider](#) est un [Window](#)

Le [Window](#) est [associé](#) [Document](#)

Si le [AnimationFrameProvider](#) est un [DedicatedWorkerGlobalScope](#)

Le [DedicatedWorkerGlobalScope](#)

Chaque [objet cible](#) a une **carte de rappels de trame d'animation** , qui est une [carte ordonnée](#) qui doit être initialement vide, et un **identificateur de rappel de trame d'animation** , qui est un nombre qui doit initialement être égal à zéro.

Un [AnimationFrameProvider](#) *fournisseur* est considéré comme **pris en charge** si l'une des conditions suivantes est remplie :

- *fournisseur* est un [Window](#).
- [Le jeu de propriétaires](#) du *fournisseur* [contient](#) un [Document](#) objet.
- Tous les [DedicatedWorkerGlobalScope](#) objets du [jeu de propriétaires](#) du *fournisseur* sont [pris en charge](#) .

Les étapes de la méthode sont : `requestAnimationFrame (callback)`

1. Si [cela](#) n'est pas [pris en charge](#) , lancez un `"NotSupportedError"` `DOMException` .
2. Soit *cible* l' [objet cible](#) de [cet](#) objet .
3. Incrémentez [l'identificateur de rappel de l'image d'animation de la](#) cible de un, et laissez *handle* être le résultat.
4. Soit *les callbacks la carte cible* des callbacks [des images d'animation](#) .
5. [Définissez](#) *callbacks [ handle ]* sur *callback* .
6. *Poignée de retour* .



Les étapes de la méthode sont : `cancelAnimationFrame (handle)`

1. Si [cela](#) n'est pas [pris en charge](#) , lancez un `"NotSupportedError"` `DOMException` .
2. Soit *callbacks* la carte des [rappels d'images d'animation](#) de [cet objet cible](#) .
3. [Supprimer](#) *les rappels [ handle ]*.

Pour **exécuter les rappels de trame d'animation** pour un [objet cible cible](#) avec un horodatage *maintenant* :

1. Soit *les callbacks la carte cible* des callbacks [des images d'animation](#) .
2. Soit *callbackHandles* le résultat de [l'obtention des clés](#) des *callbacks* .
3. [Pour chaque](#) *handle* dans *callbackHandles* , si *handle* [existe](#) dans *callbacks* :
  1. Soit *callback* des *callbacks [ handle ]*.
  2. [Supprimer](#) *les rappels [ handle ]*.
  3. [Appelez](#) *callback* , en passant *maintenant* comme seul argument, et si une exception est levée, [signalez l'exception](#) .

À l'intérieur des travailleurs, `requestAnimationFrame()` peut être utilisé avec un `OffscreenCanvas` transfert d'un [canvas](#) élément. Tout d'abord, dans le document, transférez le contrôle au travailleur :

```
const offscreenCanvas =
document.getElementById("c").transferControlToOffscreen();
worker.postMessage(offscreenCanvas, [offscreenCanvas]);
```

Ensuite, dans le worker, le code suivant va dessiner un rectangle se déplaçant de gauche à droite :

```
let ctx, pos = 0;
function draw(dt) {
  ctx.clearRect(0, 0, 100, 100);
  ctx.fillRect(pos, 0, 10, 10);
  pos += 10 * dt;
  requestAnimationFrame(draw);
}

self.onmessage = function(ev) {
  const transferredCanvas = ev.data;
  ctx = transferredCanvas.getContext("2d");
  draw();
};
```

## 1. [9 Communications](#)

### 1. [9.1 L'\[MessageEvent\]\(#\) interface](#)

## 9 Communications

L'[WebSocket](#) interface était définie ici. Il est maintenant défini dans WebSockets . [\[WEBSOCKETS\]](#)

### 9.1 L' [MessageEvent](#) interface



Les messages dans [les événements envoyés par le serveur](#) , [la messagerie inter-documents](#) , [la messagerie de canal](#) , [les canaux de diffusion](#) et les [WebSockets](#) utilisent l' [MessageEvent](#) interface pour leurs [message](#) événements : [\[WEBSOCKETS\]](#)

[Exposed=(Window,Worker,AudioWorklet)]



```
interface MessageEvent : Event {
```

```
  constructor(DOMString type, optional MessageEventInit
```

```
  eventInitDict = {});
```

```
  readonly attribute any data;
```

```
  readonly attribute USVString origin;
```

```
  readonly attribute DOMString lastEventId;
```

```
  readonly attribute MessageEventSource? source;
```

```
  readonly attribute FrozenArray<MessagePort> ports;
```

```
  undefined initMessageEvent(DOMString type, optional boolean
```

```
  bubbles = false, optional boolean cancelable = false, optional
```

```
  any data = null, optional USVString origin = "", optional
```

```
  DOMString lastEventId = "", optional MessageEventSource? source =
```

```
  null, optional sequence<MessagePort> ports = []);
```

```
};
```

```
dictionary MessageEventInit : EventInit {
```

```
  any data = null;
```

```
  USVString origin = "";
```

```
  DOMString lastEventId = "";
```

```
  MessageEventSource? source = null;
```

```
  sequence<MessagePort> ports = [];
```

```
};
```

```
typedef (WindowProxy or MessagePort or ServiceWorker)
```

```
MessageEventSource;
```

**event.data**

✓

Renvoie les données du message.

**event.origin**

✓

Renvoie l'origine du message, pour [les événements envoyés par le serveur](#) et [la messagerie interdocument](#) .

**event.lastEventId**

✓

Renvoie la [dernière chaîne d'ID d'événement](#) , pour [les événements envoyés par le serveur](#) .

**event.source**

✓

Renvoie le [WindowProxy](#) de la fenêtre source, pour [la messagerie inter-documents](#) , et le [MessagePort](#) lien, en [connect](#) cas de déclenchement sur [SharedWorkerGlobalScope](#) des objets.

**event.ports**

✓

Renvoie le [MessagePort](#) tableau envoyé avec le message, pour [la messagerie inter-documents](#) et [la messagerie de canal](#) .

L' **data** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente le message envoyé.

L' **origin** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente, dans [les événements envoyés par le serveur](#) et [la messagerie inter-documents](#) , l' [origine](#) du document qui a envoyé le message (généralement le schéma, le nom d'hôte et le port du document, mais pas son chemin ou [son fragment](#) ).

L' **lastEventId** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente, dans [les événements envoyés par le serveur](#) , la [dernière chaîne d'ID d'événement](#) de la source de l'événement.

L' **source** attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente, dans [la messagerie inter-documents](#) , le contexte [WindowProxy](#) de [navigation](#) de l' [Window](#) objet d'où provient le message ; et dans les [connect](#) événements utilisés par [les nœuds de calcul partagés](#) , la nouvelle connexion [MessagePort](#).

L' `ports` attribut doit renvoyer la valeur à laquelle il a été initialisé. Il représente, dans [la messagerie inter-documents](#) et [la messagerie de canal](#) , le `MessagePort` tableau envoyé.

La méthode doit initialiser l'événement d'une manière analogue à la méthode portant le même

nom. `[DOM].initMessageEvent(type, bubbles, cancelable, data, origin, lastEventId, source, ports).initEvent()`

*Diverses API (par exemple, `WebSocket`, `EventSource`) utilisent l' `MessageEvent` interface pour leur message événement sans utiliser l' `MessagePort` API.*

1.

## 1. [9.2 Événements envoyés par le serveur](#)

1. [9.2.1 Présentation](#)
2. [9.2.2 L' `EventSource` interface](#)
3. [9.2.3 Modèle de traitement](#)
4. [9.2.4 L' `Last-Event-ID` en-tête ``](#)
5. [9.2.5 Analyser un flux d'événements](#)
6. [9.2.6 Interprétation d'un flux d'événements](#)
7. [9.2.7 Notes de création](#)
8. [9.2.8 Push sans connexion et autres fonctionnalités](#)
9. [9.2.9 Collecte des ordures](#)
10. [9.2.10 Conseils de mise en œuvre](#)

## 9.2 Événements envoyés par le serveur



### 9.2.1 Présentation

*Cette section est non normative.*

Pour permettre aux serveurs d'envoyer des données vers des pages Web via HTTP ou à l'aide de protocoles d'envoi de serveur dédiés, cette spécification introduit l' `EventSource` interface.

L'utilisation de cette API consiste à créer un [EventSource](#) objet et à enregistrer un écouteur d'événement.

```
var source = new EventSource('updates.cgi');
source.onmessage = function (event) {
    alert(event.data);
};
```

Côté serveur, le script ("updates.cgi" dans ce cas) envoie des messages sous la forme suivante, avec le [text/event-stream](#) type MIME :

```
data : Ceci est le premier message.
```

```
data : Ceci est le deuxième message, il
data : a deux lignes.
```

```
data : C'est le troisième message.
```

---

Les auteurs peuvent séparer les événements en utilisant différents types d'événements. Voici un flux qui a deux types d'événements, "ajouter" et "supprimer" :

```
événement : ajouter
données : 73857293
```

```
événement : supprimer
données : 2153
```

```
événement : ajouter
données : 113411
```

Le script pour gérer un tel flux ressemblerait à ceci (où `addHandler` et `removeHandler` sont des fonctions qui prennent un argument, l'événement) :

```
var source = new EventSource('updates.cgi');
source.addEventListener('add', addHandler, false);
source.addEventListener('remove', removeHandler, false);
```

Le type d'événement par défaut est "message".

Les flux d'événements sont toujours décodés en UTF-8. Il n'y a aucun moyen de spécifier un autre codage de caractères.

---

Les requêtes de flux d'événements peuvent être redirigées à l'aide des redirections HTTP 301 et 307 comme pour les requêtes HTTP normales. Les clients se reconnecteront si la connexion est fermée ; un client peut être invité à arrêter de se reconnecter à l'aide du code de réponse HTTP 204 No Content.

L'utilisation de cette API plutôt que de l'émuler à l'aide [XMLHttpRequest](#) de ou [iframe](#) permet à l'agent utilisateur de faire un meilleur usage des ressources du réseau dans les cas où l'implémenteur de l'agent utilisateur et l'opérateur du réseau sont capables de se coordonner à l'avance. Entre autres avantages, cela peut entraîner des économies importantes sur la durée de vie de la batterie des appareils portables. Ceci est discuté plus en détail dans la section ci-dessous sur [le push sans connexion](#) .

## 9.2.2 L' [EventSource](#) interface



```
[Exposed=(Window,Worker)]
```

```
interface EventSource : EventTarget {
```

```
  constructor(USVString url, optional EventSourceInit
```

```
  eventSourceInitDict = {});
```

```
  readonly attribute USVString url;
```

```
  readonly attribute boolean withCredentials;
```

```
  // ready state
```

```
  const unsigned short CONNECTING = 0;
```

```
const unsigned short OPEN = 1;
```

```
const unsigned short CLOSED = 2;
```

```
readonly attribute unsigned short readyState;
```

```
// networking
```

```
attribute EventHandler onopen;
```

```
attribute EventHandler onmessage;
```

```
attribute EventHandler onerror;
```

```
undefined close();
```

```
};
```

```
dictionary EventSourceInit {
```

```
    boolean withCredentials = false;
```

```
};
```

Chaque [EventSource](#) objet est associé aux éléments suivants :

- Une **URL** (un [enregistrement d'URL](#) ). Réglé pendant la construction.
- Une **demande** . Celle-ci doit initialement être nulle.
- Un **temps de reconnexion** , en millisecondes. Il doit s'agir initialement d'une valeur [définie par l'implémentation](#) , probablement de l'ordre de quelques secondes.
- Une **dernière chaîne d'ID d'événement** . Il doit s'agir initialement de la chaîne vide.

En dehors de [l'URL](#), celles-ci ne sont pas actuellement exposées sur l' [EventSource](#) objet.

```
source = new EventSource( url [, { withCredentials: true } ] )
```

✓

Crée un nouvel [EventSource](#) objet.

*url* est une chaîne donnant l' [URL](#) qui fournira le flux d'événements.

La définition `withCredentials` sur `true` définira le [mode d'identification](#) pour les demandes de connexion à *l'url* sur "`include`".

`source.close()`

✓

Abandonne toutes les instances de l' algorithme [de récupération](#) démarrées pour cet `EventSource` objet et définit l' `readyState` attribut sur `CLOSED`.

`source.url`

✓

Renvoie l' [URL](#) fournissant le flux d'événements .

`source.withCredentials`

✓

Renvoie `true` si le [mode d'identification](#) pour les demandes de connexion à l' [URL](#) fournissant le flux d'événements est défini sur "`include`", et `false` sinon.

`source.readyState`

✓

Renvoie l'état de `EventSource` la connexion de cet objet. Il peut avoir les valeurs décrites ci-dessous.

Le constructeur, lorsqu'il est appelé, doit exécuter ces étapes :`EventSource(url, eventSourceInitDict)`

1. Soit `ev` un nouvel `EventSource` objet.
2. Soit `settings` l' [objet de paramètres pertinent](#) d' `ev` .
3. Soit `urlRecord` le résultat de [l'analyse de l'URL](#) avec l'[URL de base](#) de l'API des *paramètres* et [l'encodage des caractères de l'URL](#) de l'API des *paramètres* .
4. Si `urlRecord` est un échec, lancez un "`SyntaxError`" `DOMException` .
5. Définissez l'[URL](#) d' `ev` sur `urlRecord` .
6. Soit `corsAttributeState` être [Anonymous](#) .
7. Si la valeur du membre de `eventSourceInitDict` `withCredentials` est `true`, définissez `corsAttributeState` sur [Use Credentials](#) et définissez l' attribut `ev` `withCredentials` sur `true`.
8. Soit `request` le résultat de [la création d'une requête CORS potentielle](#) donnée `urlRecord` , la chaîne vide et `corsAttributeState` .
9. Définissez [le client](#) de *la requête* sur *paramètres* .

10. Les agents utilisateurs peuvent définir ( `Accept`, `text/event-stream` ) dans la liste d'en-tête de la *requête* .
  11. Définissez le mode de cache de la *requête* sur `".no-store`
  12. Définissez le type d'initiateur de la *requête* sur `".other`
  13. Définissez la requête d' *ev* sur *request* .
  14. Soit *processEventSourceEndOfBody* la réponse donnée *res* soit l'étape suivante : si *res* n'est pas une erreur réseau , alors rétablissez la connexion .
  15. Extraire la *requête* , avec *processResponseEndOfBody* défini sur *processEventSourceEndOfBody* et *processResponse* défini sur les étapes suivantes étant donné la réponse *res* :
    1. Si *res* est une erreur de réseau abandonnée , la connexion échoue .
    2. Sinon, si *res* est une erreur de réseau , alors rétablissez la connexion , à moins que l'agent utilisateur ne sache que cela est futile, auquel cas l'agent utilisateur peut faire échouer la connexion .
    3. Sinon, si le statut de *res* n'est pas 200, ou si `res.status` n'est pas `200` de *res* , la connexion échoue `.Content-Type: text/event-stream`
    4. Sinon, annoncez la connexion et interprétez le corps de *res* ligne par ligne.
  16. Retour *ev* .
- 

Le `url` getter de l'attribut doit retourner la sérialisation de l'url de cet `EventSource` objet .

L' `withCredentials` attribut doit renvoyer la valeur à laquelle il a été initialisé en dernier. Lorsque l'objet est créé, il doit être initialisé à `false`.

L' `readyState` attribut représente l'état de la connexion. Il peut prendre les valeurs suivantes :

#### **CONNECTING**(valeur numérique 0)

La connexion n'a pas encore été établie ou elle a été fermée et l'agent utilisateur se reconnecte.

#### **OPEN**(valeur numérique 1)

L'agent utilisateur a une connexion ouverte et distribue les événements au fur et à mesure qu'il les reçoit.



## CLOSED(valeur numérique 2)

La connexion n'est pas ouverte et l'agent utilisateur n'essaie pas de se reconnecter. Soit une erreur fatale s'est produite, soit la `close()` méthode a été invoquée.

Lorsque l'objet est créé, il `readyState` doit être défini sur `CONNECTING(0)`. Les règles données ci-dessous pour gérer la connexion définissent quand la valeur change.

La `close()` méthode doit abandonner toutes les instances de l'algorithme [de récupération](#) démarrées pour cet `EventSource` objet et doit définir l' `readyState` attribut sur `CLOSED`.

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par tous les objets implémentant l' `EventSource` interface :

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
<code>onopen</code> ✓ MDN	<code>open</code>
<code>onmessage</code> ✓ MDN	<code>message</code>
<code>onerror</code> ✓ MDN	<code>error</code>

### 9.2.3 Modèle de traitement

Lorsqu'un agent utilisateur doit **annoncer la connexion** , l'agent utilisateur doit [mettre en file d'attente une tâche](#) qui, si l' `readyState` attribut est défini sur une valeur autre que `CLOSED`, définit l' `readyState` attribut sur `OPEN` et [déclenche un événement](#) nommé `open` sur l' `EventSource` objet.

Lorsqu'un agent utilisateur doit **rétablir la connexion** , l'agent utilisateur doit exécuter les étapes suivantes. Ces étapes sont exécutées [en parallèle](#) et non dans le cadre d'une [tâche](#) . (Les tâches qu'il met en file d'attente, bien sûr, sont exécutées comme des tâches normales et non elles-mêmes [en parallèle](#) .)

1. [Mettez une tâche en file d'attente](#) pour exécuter les étapes suivantes :
  1. Si l' `readyState` attribut est défini sur `CLOSED`, abandonnez la tâche.

2. Définissez l' `readyState` attribut sur `CONNECTING`.
  3. [Lance un événement](#) nommé `error` sur l' `EventSource` objet.
2. Attendre un délai égal au temps de reconnexion de la source d'événement.
  3. En option, attendez encore un peu. En particulier, si la tentative précédente a échoué, les agents utilisateurs peuvent introduire un délai d'attente exponentiel pour éviter de surcharger un serveur potentiellement déjà surchargé. Alternativement, si le système d'exploitation a signalé qu'il n'y a pas de connectivité réseau, les agents utilisateurs peuvent attendre que le système d'exploitation annonce que la connexion réseau est rétablie avant de réessayer.
  4. Attendez que la tâche susmentionnée soit exécutée, si elle ne l'a pas encore été.
  5. [Mettez une tâche en file d'attente](#) pour exécuter les étapes suivantes :
    1. Si l' attribut `EventSource` de l'objet `readyState` n'est pas défini sur `CONNECTING`, alors retournez.
    2. Soit `request` la [requête](#) `EventSource` de l'objet .
    3. Si la [dernière chaîne d'ID d'événement](#) `EventSource` de l'objet n'est pas la chaîne vide, alors :
      1. Soit `lastEventIDValue` la [dernière chaîne d'ID d'événement](#) `EventSource` de l'objet , [encodée en UTF-8](#) .
      2. [Définissez](#) ( `Last-Event-ID` , `lastEventIDValue` ) dans [la liste d'en-tête](#) de la `requête` .
    4. [Récupérez](#) la `requête` et traitez la réponse obtenue de cette manière, le cas échéant, comme décrit précédemment dans cette section.

Lorsqu'un agent utilisateur doit faire **échouer la connexion** , l'agent utilisateur doit [mettre en file d'attente une tâche](#) qui, si l' `readyState` attribut est défini sur une valeur autre que `CLOSED`, définit l' `readyState` attribut sur `CLOSED` et [déclenche un événement](#) nommé `error` sur l' `EventSource` objet. **Une fois que l'agent utilisateur a [échoué la connexion](#) , il ne tente pas de se reconnecter.**

---

La [source de tâche](#) pour toutes [les tâches](#) mises [en file d'attente](#) par `EventSource` des objets est la **source de tâche d'événement distante** .

## 9.2.4 L' Last-Event-ID en-tête ``

L' Last-Event-ID en-tête de requête HTTP rapporte la [dernière chaîne d'ID d'événement](#) `EventSource` d'un objet au serveur lorsque l'agent utilisateur doit [rétablir la connexion](#) .

Voir [whatwg/html issue #7363](#) pour mieux définir l'espace de valeur. Il s'agit essentiellement de toute chaîne encodée en UTF-8, qui ne contient pas U+0000 NULL, U+000A LF ou U+000D CR.

## 9.2.5 Analyser un flux d'événements

Le type MIME de ce format de flux d'événements est `text/event-stream`.

Le format de flux d'événements est tel que décrit par la `stream` production de l'ABNF suivant, dont le jeu de caractères est Unicode. [\[ABNF\]](#)

```
stream      = [ bom ] *event
event       = *( comment / field ) end-of-line
comment     = colon *any-char end-of-line
field       = 1*name-char [ colon [ space ] *any-char ] end-of-line
end-of-line = ( cr lf / cr / lf )

; characters
lf          = %x000A ; U+000A LINE FEED (LF)
cr          = %x000D ; U+000D CARRIAGE RETURN (CR)
space       = %x0020 ; U+0020 SPACE
colon       = %x003A ; U+003A COLON (:)
bom         = %xFEFF ; U+FEFF BYTE ORDER MARK
name-char   = %x0000-0009 / %x000B-000C / %x000E-0039 / %x003B-10FFFF
            ; a scalar value other than U+000A LINE FEED (LF),
            U+000D CARRIAGE RETURN (CR), or U+003A COLON (:)
any-char    = %x0000-0009 / %x000B-000C / %x000E-10FFFF
            ; a scalar value other than U+000A LINE FEED (LF)
            or U+000D CARRIAGE RETURN (CR)
```

Les flux d'événements dans ce format doivent toujours être codés en UTF-8. [\[CODAGE\]](#)

Les lignes doivent être séparées par une paire de caractères U+000D RETOUR CHARIOT U+000A LINE FEED (CRLF), un seul caractère U+000A LINE FEED (LF) ou un seul caractère U+000D CARRIAGE RETURN (CR).

Étant donné que les connexions établies avec des serveurs distants pour de telles ressources sont censées être de longue durée, les UA devraient s'assurer que la mise en mémoire tampon appropriée est utilisée. En particulier, alors que la mise en mémoire tampon de ligne avec des lignes est définie pour se terminer par un seul caractère U + 000A LINE FEED (LF) est sûre, la mise en mémoire tampon de bloc ou la mise en mémoire tampon de ligne avec différentes fins de ligne attendues peut entraîner des retards dans l'envoi d'événements.

### 9.2.6 Interprétation d'un flux d'événements

Les flux doivent être décodés à l'aide de l' algorithme [de décodage UTF-8](#) .

*L' algorithme [de décodage UTF-8](#) supprime une marque d'ordre d'octet (BOM) UTF-8 de tête, le cas échéant.*

Le flux doit ensuite être analysé en lisant tout ligne par ligne, avec une paire de caractères U+000D RETOUR CHARIOT U+000A LINE FEED (CRLF), un seul caractère U+000A LINE FEED (LF) non précédé d'un U+000D CARRIAGE Le caractère RETURN (CR) et un seul caractère U+000D RETOUR CHARIOT (CR) non suivi d'un caractère U+000A LINE FEED (LF) sont les manières dont une ligne peut se terminer.

Lorsqu'un flux est analysé, un tampon *de données* , un tampon *de type d'événement* et un *dernier* tampon d'ID d'événement doivent lui être associés. Ils doivent être initialisés avec la chaîne vide.

Les lignes doivent être traitées, dans l'ordre de leur réception, comme suit :

#### **Si la ligne est vide (une ligne vide)**

[Distribuez l'événement](#) , tel que défini ci-dessous.

#### **Si la ligne commence par un caractère U+003A COLON (:)**

Ignorez la ligne.

#### **Si la ligne contient un caractère U+003A COLON (:)**

Rassemblez les caractères sur la ligne avant le premier caractère U+003A COLON (:), et laissez *field* être cette chaîne.

Rassemblez les caractères sur la ligne après le premier caractère U+003A COLON (:), et laissez *la valeur* être cette chaîne. Si *value* commence par un caractère U+0020 SPACE, supprimez-le de *value* .

[Traitez le champ](#) en suivant les étapes décrites ci-dessous, en utilisant *champ* comme nom de champ et *valeur* comme valeur de champ.

**Sinon, la chaîne n'est pas vide mais ne contient pas de caractère U+003A COLON (:)**

[Traitez le champ](#) en suivant les étapes décrites ci-dessous, en utilisant la ligne entière comme nom de champ et la chaîne vide comme valeur de champ.

Une fois la fin du fichier atteinte, toutes les données en attente doivent être supprimées. (Si le fichier se termine au milieu d'un événement, avant la dernière ligne vide, l'événement incomplet n'est pas distribué.)

---

Les étapes de **traitement du champ** avec un nom de champ et une valeur de champ dépendent du nom du champ, comme indiqué dans la liste suivante. Les noms de champs doivent être comparés littéralement, sans effectuer de pliage de casse.

**Si le nom du champ est "événement"**

Définissez le tampon *de type d'événement* sur la valeur du champ.

**Si le nom du champ est "data"**

Ajoutez la valeur du champ au tampon *de données*, puis ajoutez un seul caractère U+000A LINE FEED (LF) au tampon *de données*.

**Si le nom du champ est "id"**

Si la valeur du champ ne contient pas U+0000 NULL, définissez le *dernier* tampon d'ID d'événement sur la valeur du champ. Sinon, ignorez le champ.

**Si le nom du champ est "réessayer"**

Si la valeur du champ se compose uniquement de [chiffres ASCII](#), interprétez la valeur du champ comme un entier en base dix et définissez le [temps de reconnexion](#) du flux d'événements sur cet entier. Sinon, ignorez le champ.

**Sinon**

Le champ est ignoré.

Lorsque l'agent utilisateur doit **distribuer l'événement**, l'agent utilisateur doit traiter le tampon *de données*, le tampon *de type d'événement* et le *dernier* tampon d'ID d'événement en suivant les étapes appropriées pour l'agent utilisateur.

Pour les navigateurs Web, les étapes appropriées pour [envoyer l'événement](#) sont les suivantes :

1. Définissez la [dernière chaîne d'ID d'événement](#) de la source d'événement sur la valeur du *dernier* tampon d'ID d'événement. La mémoire tampon n'est pas réinitialisée, de sorte que la [dernière chaîne d'ID d'événement](#) de la source d'événement reste définie sur cette valeur jusqu'à la prochaine fois qu'elle est définie par le serveur.
2. Si le tampon *de données* est une chaîne vide, définissez le tampon *de données* et le tampon *de type d'événement* sur la chaîne vide et retournez.
3. Si le dernier caractère du tampon *de données* est un caractère U+000A LINE FEED (LF), supprimez le dernier caractère du tampon *de données*.
4. Soit *événement* le résultat de [la création d'un événement](#) à l'aide de `MessageEvent`, dans le [domaine pertinent](#) de l' `EventSource` objet.
5. Initialisez l'attribut de l' *événement* `type` à `message`, son `data` attribut à `data`, son `origin` attribut à la [sérialisation](#) de l' [origine](#) de l'URL finale du flux d'événements (c'est-à-dire l'URL après les redirections) et son `lastEventId` attribut à la [dernière chaîne d'ID d'événement](#) de la source de l'événement.
6. Si le tampon *de type d'événement* a une valeur autre que la chaîne vide, modifiez le [type](#) de l'événement nouvellement créé pour qu'il soit égal à la valeur du tampon *de type d'événement*.
7. Définissez le tampon *de données* et le tampon *de type d'événement* sur la chaîne vide.
8. [Mettre en file d'attente une tâche](#) qui, si l' `readyState` attribut est défini sur une valeur autre que `CLOSED`, [distribue](#) l'événement nouvellement créé à l' `EventSource` objet.

*Si un événement n'a pas de champ "id", mais qu'un événement antérieur a défini la [dernière chaîne d'ID](#) d'événement de la source de l'événement, le `lastEventId` champ de l'événement sera défini sur la valeur du dernier champ "id" vu.*

Pour les autres agents utilisateurs, les étapes appropriées pour [distribuer l'événement](#) dépendent de l'implémentation, mais au minimum, ils doivent définir les tampons *de données* et *de type d'événement* sur la chaîne vide avant de revenir.

Le flux d'événements suivant, une fois suivi d'une ligne vide :

```
données : YHOO
données : +2
données : 10
```

... entraînerait la distribution d'un événement `message` avec l'interface `MessageEvent` sur l' `EventSource` objet. L' `data` attribut de l'événement contiendrait la chaîne "YHOO\n+2\n10" (où " \n " représente une nouvelle ligne).

Cela pourrait être utilisé comme suit :

```
var stocks = new
EventSource("https://stocks.example.com/ticker.php");
stocks.onmessage = function (event) {
    var data = event.data.split('\n');
    updateStocks(data[0], data[1], data[2]);
};
```

... où `updateStocks()` est une fonction définie comme :

```
function updateStocks(symbol, delta, value) { ... }
```

... ou quelque chose comme ça.

Le flux suivant contient quatre blocs. Le premier bloc n'a qu'un commentaire et ne déclenchera rien. Le deuxième bloc a deux champs avec les noms "data" et "id" respectivement ; un événement sera déclenché pour ce bloc, avec les données "premier événement", et définira ensuite le dernier ID d'événement sur "1" de sorte que si la connexion est interrompue entre ce bloc et le suivant, le serveur recevra un `Last-Event-ID` en-tête avec la valeur `1`. Le troisième bloc déclenche un événement avec des données "deuxième événement", et a également un champ "id", cette fois sans valeur, qui réinitialise le dernier ID d'événement à la chaîne vide (ce qui signifie non `Last-Event-ID` sera désormais envoyé en cas de tentative de reconnexion). Enfin, le dernier bloc déclenche simplement un événement avec les données "troisième événement" (avec un seul caractère d'espace en tête). Notez que le dernier doit toujours se terminer par une ligne blanche, la fin du flux ne suffit pas à déclencher l'envoi du dernier événement.

```
: flux de test

données : premier événement
identifiant : 1

données:deuxième événement
identifiant

données : troisième événement
```

Le flux suivant déclenche deux événements :

```
données

données
données

données:
```

Le premier bloc déclenche des événements avec les données définies sur la chaîne vide, comme le ferait le dernier bloc s'il était suivi d'une ligne vide. Le bloc du milieu déclenche un événement avec les données définies sur un seul caractère de saut de ligne. Le dernier bloc est ignoré car il n'est pas suivi d'une ligne vide.

Le flux suivant déclenche deux événements identiques :

```
données:tester
```

```
données : test
```

C'est parce que l'espace après les deux-points est ignoré s'il est présent.

### 9.2.7 Notes de création

Les serveurs proxy hérités sont connus pour, dans certains cas, abandonner les connexions HTTP après un court délai. Pour se protéger contre de tels serveurs proxy, les auteurs peuvent inclure une ligne de commentaire (une ligne commençant par un caractère ':') toutes les 15 secondes environ.

Les auteurs souhaitant associer des connexions de sources d'événements les unes aux autres ou à des documents spécifiques précédemment servis peuvent trouver que s'appuyer sur des adresses IP ne fonctionne pas, car des clients individuels peuvent avoir plusieurs adresses IP (en raison de la présence de plusieurs serveurs proxy) et des adresses IP individuelles peuvent avoir plusieurs clients (en raison du partage d'un serveur proxy). Il est préférable d'inclure un identifiant unique dans le document lorsqu'il est servi, puis de transmettre cet identifiant dans le cadre de l'URL lorsque la connexion est établie.

Les auteurs sont également avertis que la segmentation HTTP peut avoir des effets négatifs inattendus sur la fiabilité de ce protocole, en particulier si la segmentation est effectuée par une couche différente ignorant les exigences de synchronisation. S'il s'agit d'un problème, la segmentation peut être désactivée pour servir les flux d'événements.

Les clients qui prennent en charge la limitation de connexion par serveur de HTTP peuvent rencontrer des problèmes lors de l'ouverture de plusieurs pages à partir d'un site si chaque page a un lien [EventSource](#) vers le même domaine. Les auteurs peuvent éviter cela en utilisant le mécanisme relativement complexe consistant à utiliser des noms de domaine uniques par connexion, ou en permettant à l'utilisateur d'activer ou de désactiver la [EventSource](#) fonctionnalité sur une base par page, ou en partageant un seul [EventSource](#) objet à l'aide d'un [travailleur partagé](#) .

### 9.2.8 Push sans connexion et autres fonctionnalités



Les agents utilisateurs s'exécutant dans des environnements contrôlés, par exemple des navigateurs sur des combinés mobiles liés à des opérateurs spécifiques, peuvent décharger la gestion de la connexion sur un proxy sur le réseau. Dans une telle situation, l'agent utilisateur aux fins de conformité est considéré comme incluant à la fois le logiciel du combiné et le proxy réseau.

Par exemple, un navigateur sur un appareil mobile, après avoir établi une connexion, peut détecter qu'il se trouve sur un réseau de support et demander qu'un serveur proxy sur le réseau prenne en charge la gestion de la connexion. La chronologie d'une telle situation pourrait être la suivante :

1. Le navigateur se connecte à un serveur HTTP distant et demande la ressource spécifiée par l'auteur dans le [EventSource](#) constructeur.
2. Le serveur envoie des messages occasionnels.
3. Entre deux messages, le navigateur détecte qu'il est inactif à l'exception de l'activité réseau impliquée dans le maintien de la connexion TCP, et décide de passer en mode veille pour économiser de l'énergie.
4. Le navigateur se déconnecte du serveur.
5. Le navigateur contacte un service sur le réseau et demande que le service, un "push proxy", maintienne la connexion à la place.
6. Le service "push proxy" contacte le serveur HTTP distant et demande la ressource spécifiée par l'auteur dans le [EventSource](#) constructeur (incluant éventuellement un [Last-Event-ID](#) en-tête HTTP `` , etc.).
7. Le navigateur permet à l'appareil mobile de se mettre en veille.
8. Le serveur envoie un autre message.
9. Le service "push proxy" utilise une technologie telle que OMA push pour transmettre l'événement à l'appareil mobile, qui se réveille juste assez pour traiter l'événement, puis se rendort.

Cela peut réduire l'utilisation totale des données et peut donc entraîner des économies d'énergie considérables.

Outre la mise en œuvre de l'API et [text/event-stream](#) du format de connexion existants tels que définis par la présente spécification et de manière plus distribuée comme décrit ci-dessus, les formats de cadrage d'événement définis par [d'autres spécifications applicables](#) peuvent être pris en charge. Cette spécification ne définit pas comment ils doivent être analysés ou traités.

## 9.2.9 Collecte des ordures

Alors qu'un `EventSource` objet `readyState` est `CONNECTING`, et que l'objet a un ou plusieurs écouteurs d'événements enregistrés pour les événements `open`, `message` ou `error`, il doit y avoir une référence forte de l'objet `Window` ou à partir `WorkerGlobalScope` duquel le `EventSource` constructeur de l'objet a été appelé à l' `EventSource` objet lui-même.

Alors qu'un `EventSource` objet `readyState` est `OPEN`, et que l'objet a un ou plusieurs écouteurs d'événements enregistrés pour les événements `message` ou `error`, il doit y avoir une référence forte de l' objet `Window` ou à partir `WorkerGlobalScope` duquel le `EventSource` constructeur de l'objet a été appelé à l' `EventSource` objet lui-même.

Lorsqu'une tâche est mise en file d'attente par un `EventSource` objet sur la [source de tâche d'événement distante](#) , il doit exister une référence forte de l' objet `Window` ou `WorkerGlobalScope` à partir duquel le `EventSource` constructeur de l'objet a été appelé à cet `EventSource` objet.

Si un agent utilisateur doit **fermer** un `EventSource` objet de force (cela se produit lorsqu'un `Document` objet disparaît définitivement), l'agent utilisateur doit abandonner toutes les instances de l' algorithme [de récupération](#) démarrées pour cet `EventSource` objet et doit définir l' `readyState` attribut sur `CLOSED`.

Si un `EventSource` objet est ramassé alors que sa connexion est encore ouverte, l'agent utilisateur doit abandonner toute instance de l' algorithme [de récupération](#) ouverte par this `EventSource`.

### 9.2.10 Conseils de mise en œuvre

*Cette section est non normative.*

Les agents utilisateurs sont fortement invités à fournir des informations de diagnostic détaillées sur `EventSource` les objets et leurs connexions réseau associées dans leurs consoles de développement, afin d'aider les auteurs à déboguer le code à l'aide de cette API.

Par exemple, un agent utilisateur pourrait avoir un panneau affichant tous les `EventSource` objets qu'une page a créés, chacun listant les arguments du constructeur, s'il y a eu une erreur réseau, quel est le statut CORS de la connexion et quels en-têtes ont été envoyés par le client et reçus du serveur pour conduire à cet état, les messages qui ont été reçus et comment ils ont été analysés, et ainsi de suite.

Les implémentations sont particulièrement encouragées à signaler des informations détaillées à leurs consoles de développement chaque fois qu'un `error` événement est déclenché, car peu ou pas d'informations peuvent être mises à disposition dans les événements eux-mêmes.

1.

1. [9.3 Messagerie inter-documents](#)
  1. [9.3.1 Présentation](#)
  2. [9.3.2 Sécurité](#)
    1. [9.3.2.1 Auteurs](#)
    2. [9.3.2.2 Agents utilisateurs](#)
  3. [9.3.3 Publier des messages](#)
2. [9.4 Messagerie du canal](#)
  1. [9.4.1 Présentation](#)
    1. [9.4.1.1 Exemples](#)
    2. [9.4.1.2 Les ports comme base d'un modèle de capacité objet sur le Web](#)
    3. [9.4.1.3 Ports comme base de l'abstraction des implémentations de service](#)
  2. [9.4.2 Canaux de messages](#)
  3. [9.4.3 Ports de messages](#)
  4. [9.4.4 Diffusion vers de nombreux ports](#)
  5. [9.4.5 Ports et collecte des ordures](#)
3. [9.5 Diffusion vers d'autres contextes de navigation](#)

## 9.3 Messagerie inter-documents



Les navigateurs Web, pour des raisons de sécurité et de confidentialité, empêchent les documents de différents domaines de s'affecter les uns les autres ; c'est-à-dire que les scripts intersites ne sont pas autorisés.

Bien qu'il s'agisse d'une fonctionnalité de sécurité importante, elle empêche les pages de différents domaines de communiquer même lorsque ces pages ne sont pas hostiles. Cette section présente un système de messagerie qui permet aux documents de communiquer entre eux quel que soit leur domaine source, d'une manière conçue pour ne pas permettre les attaques de script intersite.

*L' [postMessage\(\)](#) API peut être utilisée comme [vecteur de suivi](#) .*

### 9.3.1 Présentation

*Cette section est non normative.*

Par exemple, si le document A contient un `iframe` élément qui contient le document B et que le script du document A appelle `postMessage()` l' `Window` objet du document B, un événement de message sera déclenché sur cet objet, marqué comme provenant du `Window` document A. Le script dans le document A pourrait ressembler à :

```
var o = document.getElementsByTagName('iframe')[0];
o.contentWindow.postMessage('Hello world',
'https://b.example.org/');
```

Pour enregistrer un gestionnaire d'événements pour les événements entrants, le script utiliserait `addEventListener()` (ou des mécanismes similaires). Par exemple, le script du document B pourrait ressembler à :

```
window.addEventListener('message', receiver, false);
function receiver(e) {
  if (e.origin == 'https://example.com') {
    if (e.data == 'Hello world') {
      e.source.postMessage('Hello', e.origin);
    } else {
      alert(e.data);
    }
  }
}
```

Ce script vérifie d'abord que le domaine est le domaine attendu, puis examine le message, qu'il affiche soit à l'utilisateur, soit auquel il répond en renvoyant un message au document qui a envoyé le message en premier lieu.

## 9.3.2 Sécurité

### 9.3.2.1 Auteurs

***L'utilisation de cette API nécessite une attention particulière pour protéger les utilisateurs des entités hostiles qui abusent d'un site à leurs propres fins.***

Les auteurs doivent vérifier l' `origin` attribut pour s'assurer que les messages ne sont acceptés que des domaines dont ils s'attendent à recevoir des messages. Sinon, des bogues dans le code de gestion des messages de l'auteur pourraient être exploités par des sites hostiles.

De plus, même après avoir vérifié l' `origin` attribut, les auteurs doivent également vérifier que les données en question sont au format attendu. Sinon, si la source de l'événement a été attaquée à l'aide d'une faille de script intersite, un traitement

supplémentaire non contrôlé des informations envoyées à l'aide de la `postMessage()` méthode pourrait entraîner la propagation de l'attaque dans le récepteur.

Les auteurs ne doivent pas utiliser le mot-clé générique (\*) dans l'argument `targetOrigin` dans les messages contenant des informations confidentielles, sinon il n'y a aucun moyen de garantir que le message est uniquement remis au destinataire auquel il était destiné.

---

Les auteurs qui acceptent des messages de n'importe quelle origine sont encouragés à considérer les risques d'une attaque par déni de service. Un attaquant pourrait envoyer un volume élevé de messages ; si la page réceptrice effectue des calculs coûteux ou provoque l'envoi de trafic réseau pour chacun de ces messages, le message de l'attaquant pourrait être multiplié en une attaque par déni de service. Les auteurs sont encouragés à utiliser la limitation de débit (n'acceptant qu'un certain nombre de messages par minute) pour rendre de telles attaques impossibles.

### 9.3.2.2 Agents utilisateurs

L'intégrité de cette API repose sur l'incapacité pour les scripts d'une [origine](#) de publier des événements arbitraires (en utilisant `dispatchEvent()` ou non) sur des objets d'autres origines (ceux qui ne sont pas les [mêmes](#)).

*Les responsables de la mise en œuvre sont invités à prendre des précautions supplémentaires lors de la mise en œuvre de cette fonctionnalité. Il permet aux auteurs de transmettre des informations d'un domaine à un autre domaine, ce qui est normalement interdit pour des raisons de sécurité. Cela exige également que les UA veillent à autoriser l'accès à certaines propriétés mais pas à d'autres.*

---

Les agents utilisateurs sont également encouragés à envisager de limiter le trafic des messages entre différentes [origines](#), afin de protéger les sites naïfs contre les attaques par déni de service.

### 9.3.3 Publier des messages

```
window.postMessage(message [, options ])
```



Affiche un message dans la fenêtre donnée. Les messages peuvent être des objets structurés, par exemple des objets imbriqués et des tableaux, peuvent contenir des valeurs JavaScript (chaînes, nombres, [Date](#) objets, etc.) et peuvent contenir certains objets de données tels que [File Blob](#), [FileList](#) et [ArrayBuffer](#) des objets.

Les objets listés dans le [transfer](#) membre des *options* sont transférés, pas seulement clonés, ce qui signifie qu'ils ne sont plus utilisables côté envoi.

Une origine cible peut être spécifiée à l'aide du [targetOrigin](#) membre de *options*. S'il n'est pas fourni, la valeur par défaut est `" /"`. Cette valeur par défaut limite le message aux cibles de même origine uniquement.

Si l'origine de la fenêtre cible ne correspond pas à l'origine cible donnée, le message est ignoré pour éviter les fuites d'informations. Pour envoyer le message à la cible quelle que soit son origine, définissez l'origine de la cible sur `" *"`.

Lève un ["DataCloneError"](#) [DOMException](#) si le tableau de transfert contient des objets en double ou si le message n'a pas pu être cloné.

```
window.postMessage(message, targetOrigin [, transfer ])
```

Il s'agit d'une version alternative de [postMessage\(\)](#) l'endroit où l'origine cible est spécifiée en tant que paramètre. L'appel `window.postMessage(message, target, transfer)` est équivalent à `window.postMessage(message, {targetOrigin, transfer})`.

Lorsque la publication d'un message dans [Window](#) un [contexte de navigation](#) qui vient d'être navigué vers un nouveau [Document](#) est susceptible d'avoir pour conséquence que le message ne reçoive pas son destinataire : les scripts dans le [contexte de navigation](#) cible doivent avoir eu le temps de configurer des écouteurs pour les messages. Ainsi, par exemple, dans les situations où un message doit être envoyé à l'[Window](#) enfant nouvellement créé [iframe](#), il est conseillé aux auteurs de faire en sorte que l'enfant [Document](#) poste un message à son parent annonçant qu'il est prêt à recevoir des messages, et que le parent attende ce message avant de commencer à poster des messages.

Les **étapes de message de la fenêtre de publication**, étant donné une *targetWindow*, un message et des options, sont les suivantes :

1. Soit *targetRealm* le [domaine](#) de *targetWindow*.
2. Soit *incumbentSettings* l'[objet de paramètres incumbent](#).
3. Soit *targetOrigin* les options [`" targetOrigin"`].
4. Si *targetOrigin* est un seul caractère SOLIDUS `U+002F` (`/`), définissez *targetOrigin* sur *incumbentSettings* *origin*.

5. Sinon, si *targetOrigin* n'est pas un seul caractère U+002A ASTÉRISQUE (\*), alors :
  1. Soit *parsedURL* le résultat de l'exécution de [l'analyseur d'URL](#) sur *targetOrigin* .
  2. Si *parsedURL* est un échec, lancez un ["SyntaxError" DOMException](#) .
  3. Définissez *targetOrigin* sur [l'origine](#) de *parsedURL* .
6. Soit *transfert* les options [[" transfer"](#)].
7. Laissez *serializeWithTransferResult* être [StructuredSerializeWithTransfer](#) ( *message* , *transfer* ). Renvoyez toutes les exceptions.
8. [Mettez en file d'attente une tâche globale](#) sur la **source de tâche de message publiée** donnée *targetWindow* pour exécuter les étapes suivantes :
  1. Si l' argument *targetOrigin* n'est pas un seul caractère littéral U+002A ASTERISK (\*) et que l' [origine](#) de *targetWindow* [associée Document](#) n'est pas [la même origine](#) que *targetOrigin* , alors retournez.
  2. Soit *origin* la [sérialisation](#) de [l'origine](#) de *incumbentSettings* .
  3. Soit *source* l' [WindowProxy](#) objet correspondant à [l'objet global](#) de *incumbentSettings* (un objet).[Window](#)
  4. Laissez *deserializeRecord* être [StructuredDeserializeWithTransfer](#) ( *serializeWithTransferResult* , *targetRealm* ).

Si cela lève une exception, attrapez-la, [déclenchez un événement](#) nommé [messageerror](#) à *targetWindow* , en utilisant [MessageEvent](#), avec l' [origin](#) attribut initialisé à *origin* et l' [source](#) attribut initialisé à *source* , puis revenez.

  5. Laissez *messageClone* être *deserializeRecord* .[[Deserialized]].
  6. Soit *newPorts* un nouveau [tableau figé](#) composé de tous [MessagePort](#) les objets dans *deserializeRecord* .[[TransferredValues]], le cas échéant, en conservant leur ordre relatif.
  7. [Déclenchez un événement](#) nommé [message](#) à *targetWindow* , en utilisant [MessageEvent](#), avec l' [origin](#) attribut initialisé à *origin* , l' [source](#) attribut initialisé à *source* , l' [data](#) attribut initialisé à *messageClone* et l' [ports](#) attribut initialisé à *newPorts* .

Les étapes de la méthode [Window](#) de l'interface consistent à exécuter les [étapes de publication de message de la fenêtre](#) en fonction de [this](#) , *message* et *options* .[postMessage\(message, options\)](#)

Les étapes `Window` de la méthode de l'interface consistent à exécuter les [étapes de message de la fenêtre de publication](#) étant donné [this](#) , `message` , et «[ " " → `targetOrigin` , " " → `transfer` ]».`postMessage(message, targetOrigin, transfer)`[targetOrigin](#)[transfer](#)

## 9.4 Messagerie du canal



### 9.4.1 Présentation

*Cette section est non normative.*

Pour permettre à des morceaux de code indépendants (ex. s'exécutant dans différents [contextes de navigation](#) ) de communiquer directement, les auteurs peuvent utiliser [la messagerie de canal](#) .

Les canaux de communication dans ce mécanisme sont mis en œuvre sous forme de canaux bidirectionnels, avec un port à chaque extrémité. Les messages envoyés dans un port sont livrés à l'autre port, et vice-versa. Les messages sont livrés sous forme d'événements DOM, sans interrompre ni bloquer [les tâches](#) en cours d'exécution .

Pour créer une connexion (deux ports "intriqués"), le `MessageChannel()` constructeur est appelé :

```
var channel = new MessageChannel();
```

L'un des ports est conservé comme port local, et l'autre port est envoyé au code distant, par exemple en utilisant `postMessage()` :

```
otherWindow.postMessage('hello', 'https://example.com',  
[channel.port2]);
```

Pour envoyer des messages, la `postMessage()` méthode sur le port est utilisée :

```
channel.port1.postMessage('hello');
```

Pour recevoir des messages, on écoute les [message](#) événements :

```
channel.port1.onmessage = handleMessage;  
function handleMessage(event) {
```



```
// message is in event.data
// ...
}
```

Les données envoyées sur un port peuvent être des données structurées ; par exemple ici un tableau de chaînes est passé sur a [MessagePort](#):

```
port1.postMessage(['hello', 'world']);
```

### 9.4.1.1 Exemples

*Cette section est non normative.*

Dans cet exemple, deux bibliothèques JavaScript sont connectées l'une à l'autre à l'aide [MessagePort](#) de s. Cela permet aux bibliothèques d'être hébergées ultérieurement dans différents cadres, ou dans [Worker](#) des objets, sans aucune modification des API.

```
<script src="contacts.js"></script> <!-- exposes a contacts object
-->

<script src="compose-mail.js"></script> <!-- exposes a composer
object -->

<script>
  var channel = new MessageChannel();
  composer.addContactsProvider(channel.port1);
  contacts.registerConsumer(channel.port2);
</script>
```

Voici à quoi pourrait ressembler l'implémentation de la fonction "addContactsProvider()" :

```
function addContactsProvider(port) {
  port.onmessage = function (event) {
    switch (event.data.messageType) {
      case 'search-result': handleSearchResult(event.data.results);
      break;
      case 'search-done': handleSearchDone(); break;
      case 'search-error': handleSearchError(event.data.message);
      break;
      // ...
    }
  };
};
```

Alternativement, il pourrait être mis en œuvre comme suit :

```
function addContactsProvider(port) {
  port.addEventListener('message', function (event) {
    if (event.data.messageType == 'search-result')
      handleSearchResult(event.data.results);
  });
  port.addEventListener('message', function (event) {
    if (event.data.messageType == 'search-done')
      handleSearchDone();
  });
  port.addEventListener('message', function (event) {
    if (event.data.messageType == 'search-error')
      handleSearchError(event.data.message);
  });
  // ...
  port.start();
};
```

La principale différence est que lors de l'utilisation de `addEventListener()`, la `start()` méthode doit également être invoquée. Lors de l'utilisation de `onmessage`, l'appel à `start()` est implicite.

La `start()` méthode, qu'elle soit appelée explicitement ou implicitement (en définissant `onmessage`), démarre le flux de messages : les messages postés sur les ports de messages sont initialement mis en pause, de sorte qu'ils ne tombent pas par terre avant que le script n'ait eu la chance de configurer son manutentionnaires.

#### 9.4.1.2 Les ports comme base d'un modèle de capacité objet sur le Web

*Cette section est non normative.*

Les ports peuvent être considérés comme un moyen d'exposer des capacités limitées (au sens du modèle de capacité d'objet) à d'autres acteurs du système. Il peut s'agir soit d'un système à faible capacité, où les ports sont simplement utilisés comme un modèle pratique au sein d'une origine particulière, soit d'un modèle à forte capacité, où ils sont fournis par un fournisseur d'origine comme le seul mécanisme par lequel un autre consommateur d'origine peut effectuer changer de fournisseur ou obtenir des informations auprès de celui-ci.

Par exemple, considérons une situation dans laquelle un site Web social intègre dans un `iframe` le fournisseur de contacts de messagerie de l'utilisateur (un site de carnet d'adresses, d'une deuxième origine), et dans un second `iframe` un jeu (d'une

troisième origine). Le site social externe et le jeu du second `iframe` ne peuvent accéder à rien à l'intérieur du premier `iframe`; ensemble, ils ne peuvent que :

- [Naviguez](#) dans le `iframe` vers une nouvelle [URL](#) , telle que la même [URL](#) mais avec un [fragment](#) différent , ce qui fait que le `Window` dans le `iframe` reçoit un `hashchange` événement.
- Redimensionnez le `iframe`, ce qui fait que le `Window` dans `iframe` reçoit un `resize` événement.
- Envoyez un `message` événement à dans `Window` à `iframe` l'aide de l' `window.postMessage()` API.

Le fournisseur de contacts peut utiliser ces méthodes, plus particulièrement la troisième, pour fournir une API accessible par d'autres origines pour manipuler le carnet d'adresses de l'utilisateur. Par exemple, il pourrait répondre à un message `"add-contact Guillaume Tell <tell@pomme.example.net>"` en ajoutant la personne et l'adresse e-mail données au carnet d'adresses de l'utilisateur.

Pour éviter qu'un site Web ne puisse manipuler les contacts de l'utilisateur, le fournisseur de contacts peut n'autoriser que certains sites de confiance, tels que le site social, à le faire.

Supposons maintenant que le jeu veuille ajouter un contact au carnet d'adresses de l'utilisateur et que le site social soit disposé à lui permettre de le faire en son nom, essentiellement en "partageant" la confiance que le fournisseur de contacts avait avec le site social. Il y a plusieurs façons de le faire ; plus simplement, il pourrait simplement s'agir de messages proxy entre le site du jeu et le site des contacts. Cependant, cette solution présente un certain nombre de difficultés : elle oblige le site social à faire entièrement confiance au site de jeu pour ne pas abuser du privilège, ou elle exige que le site social vérifie chaque demande pour s'assurer qu'il ne s'agit pas d'une demande qu'il ne fait pas. souhaitez autoriser (par exemple, ajouter plusieurs contacts, lire les contacts ou les supprimer) ; cela nécessite également une complexité supplémentaire s'il y a '

Cependant, en utilisant des canaux de messages et `MessagePort` des objets, tous ces problèmes peuvent disparaître. Lorsque le jeu indique au site social qu'il souhaite ajouter un contact, le site social peut demander au fournisseur de contacts non pas d'ajouter un contact, mais la *possibilité* d'ajouter un seul contact. Le fournisseur de contacts crée alors une paire d' `MessagePort` objets, et renvoie l'un d'entre eux au site social, qui le transmet au jeu. Le jeu et le fournisseur de contacts ont alors une connexion directe, et le fournisseur de contacts sait qu'il n'honore qu'une seule demande "ajouter un contact", rien d'autre. En d'autres termes, le jeu a la possibilité d'ajouter un seul contact.

#### 9.4.1.3 Ports comme base de l'abstraction des implémentations de service

*Cette section est non normative.*

Poursuivant l'exemple de la section précédente, considérez le fournisseur de contacts en particulier. Alors qu'une implémentation initiale aurait pu simplement utiliser [XMLHttpRequest](#) des objets dans le service [iframe](#), une évolution du service pourrait plutôt vouloir utiliser un [travailleur partagé](#) avec une seule [WebSocket](#) connexion.

Si la conception initiale utilisait [MessagePort](#) des objets pour accorder des capacités, ou même simplement pour autoriser plusieurs sessions indépendantes simultanées, l'implémentation du service peut passer du modèle [XMLHttpRequests-in-each-iframe](#) au [WebSocket](#) modèle partagé sans changer du tout l'API : les ports sur le côté fournisseur de services peuvent tous être transmis au travailleur partagé sans que cela n'affecte le moins du monde les utilisateurs de l'API.

## 9.4.2 Canaux de messages

✓ MDN

```
[Exposed=(Window,Worker)]
```

```
interface MessageChannel {
```

```
    constructor();
```

```
    readonly attribute MessagePort port1;
```

```
    readonly attribute MessagePort port2;
```

```
};
```

```
channel = new MessageChannel()
```

✓

Renvoie un nouvel [MessageChannel](#) objet avec deux nouveaux [MessagePort](#) objets.

```
channel.port1
```

✓

Renvoie le premier [MessagePort](#) objet.

```
channel.port2
```

✓

Renvoie le deuxième [MessagePort](#) objet.

Un `MessageChannel` objet a un **port 1 associé et un port 2** associé , tous deux `MessagePort` objets.

Les `new MessageChannel()` étapes du constructeur sont :

1. Définissez ce port 1 sur un nouveau `MessagePort` dans le domaine concerné .
2. Définissez ce port 2 sur un nouveau `MessagePort` dans le domaine concerné .
3. Enchevêtrez ce port 1 et ce port 2 .

Les `port1` étapes du getter consistent à renvoyer ce port 1 .

Les `port2` étapes du getter consistent à retourner ce port 2 .

### 9.4.3 Ports de messages



Chaque canal a deux ports de messages. Les données envoyées via un port sont reçues par l'autre port, et vice versa.

```
[Exposed=(Window,Worker,AudioWorklet), Transferable]
```

```
interface MessagePort : EventTarget {
```

```
    undefined postMessage(any message, sequence<object> transfer);
```

```
    undefined postMessage(any message, optional
```

```
    StructuredSerializeOptions options = {});
```

```
    undefined start();
```

```
    undefined close();
```

```
// event handlers
```

```
    attribute EventHandler onmessage;
```

```
    attribute EventHandler onmessageerror;
```

```
};
```

```
dictionary StructuredSerializeOptions {
```

```
sequence<object> transfer = [];
```

```
};
```

```
port.postMessage(message [, transfer])
```

✓

```
port.postMessage(message [, { transfer }])
```

Publie un message via le canal. Les objets répertoriés dans *le transfert* sont transférés, pas seulement clonés, ce qui signifie qu'ils ne sont plus utilisables du côté de l'envoi.

Lève un "[DataCloneError](#)" [DOMException](#) si *le transfert* contient des objets ou un *port* en double , ou si *le message* n'a pas pu être cloné.

```
port.start()
```

✓

Commence à répartir les messages reçus sur le port.

```
port.close()
```

✓

Déconnecte le port, de sorte qu'il n'est plus actif.

Chaque [MessagePort](#) objet peut être intriqué avec un autre (une relation symétrique). Chaque [MessagePort](#) objet possède également une [source de tâche](#) appelée la **file d'attente de messages du port** , initialement vide. Une [file d'attente de messages de port](#) peut être activée ou désactivée et est initialement désactivée. Une fois activé, un port ne peut plus jamais être désactivé (bien que les messages de la file d'attente puissent être déplacés vers une autre file d'attente ou supprimés complètement, ce qui a à peu près le même effet). A [MessagePort](#) a également un indicateur **a été expédié** , qui doit initialement être faux.

Lorsque la [file d'attente de messages du port](#) d'un port est activée, la [boucle d'événements](#) doit l'utiliser comme l'une de ses [sources de tâches](#) . Lorsque l'[objet global pertinent](#) d'un port est un [Window](#), toutes [les tâches mises en file d'attente](#) sur sa [file d'attente de messages de port](#) doivent être associées à l' [objet global pertinent](#) du port [associé à Document](#) .

*Si le document est [entièrement actif](#) , mais que les écouteurs d'événements ont tous été créés dans le contexte de documents qui ne sont pas [entièrement actifs](#) , les messages ne seront pas reçus à moins que et jusqu'à ce que les documents redeviennent [entièrement actifs](#) .*

Chaque [boucle d'événements](#) a une [source de tâche](#) appelée la **file d'attente des messages du port non livré** . Il s'agit d'une [source de tâches](#) virtuelle : elle doit agir comme si elle contenait les [tâches](#) de chaque [file d'attente de messages du port](#) de

chacun `MessagePort` dont l'indicateur a été expédié est faux, dont la file d'attente de messages du port est activée et dont la boucle d'événements de l'agent concerné est cette boucle d'événements, dans l'ordre dans lequel ils ont été ajoutés à leur source de tâche respective. Lorsqu'une tâche serait supprimée de la file d'attente des messages du port non expédié, il doit à la place être retiré de sa file d'attente de messages de port.

Lorsqu'un `MessagePort` indiqueur a été expédié est faux, sa file d'attente de messages de port doit être ignorée pour les besoins de la boucle d'événements. (La file d'attente des messages du port non expédié est utilisée à la place.)

*L'indicateur a été expédié est défini sur vrai lorsqu'un port, son jumeau ou l'objet à partir duquel il a été cloné est ou a été transféré. Lorsqu'un `MessagePort` indiqueur a été expédié est vrai, sa file d'attente de messages de port agit comme une source de tâche de première classe, non affectée par une file d'attente de messages de port non expédiée.*

Lorsque l'agent utilisateur doit **intriquer** deux `MessagePort` objets, il doit exécuter les étapes suivantes :

1. Si l'un des ports est déjà enchevêtré, démêlez-le et le port avec lequel il était enchevêtré.

*Si ces deux ports précédemment intriqués étaient les deux ports d'un `MessageChannel` objet, alors cet `MessageChannel` objet ne représente plus un canal réel : les deux ports de cet objet ne sont plus intriqués.*

2. Associez les deux ports à enchevêtrer, afin qu'ils forment les deux parties d'un nouveau canal. (Aucun `MessageChannel` objet ne représente ce canal.)

Deux ports A et B qui sont passés par cette étape sont désormais dits intriqués ; l'un est intriqué dans l'autre, et vice versa.

*Bien que cette spécification décrive ce processus comme instantané, les implémentations sont plus susceptibles de l'implémenter via la transmission de messages. Comme pour tous les algorithmes, la clé est "simplement" que le résultat final soit indiscernable, dans un sens de boîte noire, de la spécification.*

---

`MessagePort` les objets sont des objets transférables. Leurs étapes de transfert, compte tenu de *value* et *dataHolder*, sont :

1. Définir la valeur's a été expédié comme indicateur sur true.

2. Définissez `dataHolder` .[[PortMessageQueue]] sur [la file d'attente de messages](#) du port de *la valeur* .
3. Si *la valeur* est intriquée avec un autre port *remotePort* , alors :
  1. Définissez l' indicateur *remotePort* [sur](#) `true`.
  2. Définissez `dataHolder` .[[RemotePort]] sur *remotePort* .
4. Sinon, définissez `dataHolder` .[[RemotePort]] sur `null`.

Leurs [étapes de réception de transfert](#) , étant donné *dataHolder* et *value* , sont :

1. Définir *la valeur* 's [a été expédié](#) comme indicateur sur `true`.
2. Déplacez toutes les [tâches](#) qui doivent déclencher [message](#) des événements dans `dataHolder` .[[PortMessageQueue]] vers la [file d'attente de messages de port](#) de *value* , le cas échéant, en laissant [la file d'attente de messages de port](#) de *value* dans son état initial désactivé et, si *value* ' l' [objet global pertinent](#) est un , associant les [tâches](#) déplacées à l' objet [global pertinent associé](#) à la *valeur* .[WindowDocument](#)
3. Si `dataHolder` .[[RemotePort]] n'est pas nul, alors [associez](#) `dataHolder` .[[RemotePort]] et *value* . (Cela séparera `dataHolder` .[[RemotePort]] du port d'origine qui a été transféré.)

---

Les **étapes de message du port de message** , étant donné *sourcePort* , *targetPort* , *message* et *les options* sont les suivantes :

1. Soit *transfert* les *options* [" [transfer](#)"].
2. Si *transfer* [contient](#) *sourcePort* , lancez un "[DataCloneError](#)" [DOMException](#) .
3. Que *doomed* soit faux.
4. Si *targetPort* n'est pas null et que *transfer* [contient](#) *targetPort* , définissez *doomed* sur `true` et signalez éventuellement à une console de développement que le port cible a été publié sur lui-même, entraînant la perte du canal de communication.
5. Laissez *serializeWithTransferResult* être [StructuredSerializeWithTransfer](#) ( *message* , *transfer* ). Renvoyez toutes les exceptions.
6. Si *targetPort* est null, ou si *doomed* est vrai, alors retournez.



7. Ajoutez une [tâche](#) qui exécute les étapes suivantes à la [file d'attente des messages du port](#) de `targetPort` :

1. Soit `finalTargetPort` la file [d'attente de messages du port](#) `MessagePort` dans laquelle la tâche se trouve maintenant.

*Cela peut être différent de `targetPort` , si `targetPort` lui-même a été transféré et donc toutes ses tâches déplacées avec lui.*

2. Soit `targetRealm` le [domaine pertinent](#) de `finalTargetPort` .
3. Laissez `deserializeRecord` être [StructuredDeserializeWithTransfer](#) ( `serializeWithTransferResult` , `targetRealm` ).

Si cela lève une exception, attrapez-la, [déclenchez un événement](#) nommé `messageerror` à `finalTargetPort` , en utilisant `MessageEvent`, puis revenez.

4. Laissez `messageClone` être `deserializeRecord` .[[Deserialized]].
5. Soit `newPorts` un nouveau [tableau figé](#) composé de tous `MessagePort` les objets dans `deserializeRecord` .[[TransferredValues]], le cas échéant, en conservant leur ordre relatif.
6. [Déclenchez un événement](#) nommé `message` à `finalTargetPort` , en utilisant `MessageEvent`, avec l' `data` attribut initialisé à `messageClone` et l' `ports` attribut initialisé à `newPorts` .

Les étapes de la méthode sont : `postMessage(message, options)`

1. Soit `targetPort` le port avec lequel [ceci](#) est enchevêtré, le cas échéant ; sinon laissez-le être nul.
2. Exécutez les [étapes de publication de message du port de message](#) en fournissant [this](#) , `targetPort` , `message` et `options` .

Les étapes de la méthode sont : `postMessage(message, transfer)`

1. Soit `targetPort` le port avec lequel [ceci](#) est enchevêtré, le cas échéant ; sinon laissez-le être nul.
  2. Soit les `options` «[ " `transfer` " → `transfert` ]».
  3. Exécutez les [étapes de publication de message du port de message](#) en fournissant [this](#) , `targetPort` , `message` et `options` .
-

Les `start()` étapes de la méthode consistent à activer la file d'attente de messages de [ce port](#) , si elle n'est pas déjà activée.

---

Les `close()` étapes de la méthode sont :

1. Définissez la valeur de [cet](#) emplacement interne [\[\[Detached\]\]](#) sur `true`.
  2. Si [celui-ci](#) est emmêlé, démêlez-le.
- 

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par tous les objets implémentant l' [MessagePort](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<code>onmessage</code> ✓ MDN	<a href="#">message</a>
<code>onmessageerror</code> MDN	<a href="#">messageerror</a>

La première fois que l' attribut IDL [MessagePort](#) d'un objet `onmessage` est défini, la [file d'attente de messages du port](#) doit être activée, comme si la `start()` méthode avait été appelée.

#### 9.4.4 Diffusion vers de nombreux ports

*Cette section est non normative.*

La diffusion vers de nombreux ports est en principe relativement simple : conservez un tableau d' [MessagePort](#) objets auxquels envoyer des messages et parcourez le tableau pour envoyer un message. Cependant, cela a un effet plutôt malheureux : cela empêche les ports d'être ramassés, même si l'autre côté est parti. Pour éviter ce problème, implémentez un protocole simple par lequel l'autre côté reconnaît qu'il existe toujours. S'il ne le fait pas après un certain temps, supposez qu'il est parti, fermez l' [MessagePort](#) objet et laissez-le être ramassé.

### 9.4.5 Ports et collecte des ordures

Lorsqu'un `MessagePort` objet `o` est intriqué, les agents utilisateurs doivent soit agir comme si l'objet intriqué de `o` avait une référence forte à `o`, soit comme si [l'objet global pertinent](#) de `o` avait une référence forte à `o`.

*Ainsi, un port de message peut être reçu, donné un écouteur d'événement, puis oublié, et tant que cet écouteur d'événement peut recevoir un message, le canal sera maintenu.*

*Bien sûr, si cela devait se produire des deux côtés du canal, les deux ports pourraient être ramassés, car ils ne seraient pas accessibles à partir du code en direct, malgré une forte référence l'un à l'autre.*

De plus, un `MessagePort` objet ne doit pas faire l'objet d'un ramasse-miettes tant qu'il existe un événement référencé par une [tâche](#) dans une [file d'attente de tâches](#) qui doit être distribuée sur cet `MessagePort` objet, ou tant que la [file d'attente de messages du port](#) `MessagePort` de l'objet est activée et non vide.

*Les auteurs sont fortement encouragés à fermer explicitement `MessagePort` les objets pour les démêler, afin que leurs ressources puissent être récupérées. La création de nombreux `MessagePort` objets et leur suppression sans les fermer peut entraîner une utilisation élevée de la mémoire transitoire, car la récupération de place n'est pas nécessairement effectuée rapidement, en particulier pour les `MessagePort`s où la récupération de place peut impliquer une coordination inter-processus.*

## 9.5 Diffusion vers d'autres contextes de navigation



Les pages d'une même [origine](#) ouvertes par le même utilisateur dans le même agent utilisateur mais dans différents [contextes de navigation](#) non liés doivent parfois s'envoyer des notifications, par exemple "hé, l'utilisateur connecté ici, vérifiez à nouveau vos informations d'identification".

Pour les cas élaborés, par exemple pour gérer le verrouillage de l'état partagé, pour gérer la synchronisation des ressources entre un serveur et plusieurs clients locaux, pour partager une `WebSocket` connexion avec un hôte distant, etc., [les travailleurs partagés](#) sont la solution la plus appropriée.

Pour les cas simples, cependant, où un travailleur partagé représenterait une surcharge déraisonnable, les auteurs peuvent utiliser le simple mécanisme de diffusion basé sur les canaux décrit dans cette section.

```
[Exposed=(Window,Worker)]
```

```
interface BroadcastChannel : EventTarget {
```

```
  constructor(DOMString name);
```

```
  readonly attribute DOMString name;
```

```
  undefined postMessage(any message);
```

```
  undefined close();
```

```
  attribute EventHandler onmessage;
```

```
  attribute EventHandler onmessageerror;
```

```
};
```

```
broadcastChannel = new BroadcastChannel(name)
```

✓

Renvoie un nouvel [BroadcastChannel](#) objet via lequel les messages pour le nom de canal donné peuvent être envoyés et reçus.

```
broadcastChannel.name
```

✓

Renvoie le nom du canal (tel qu'il est passé au constructeur).

```
broadcastChannel.postMessage(message)
```

✓

Envoie le message donné à d'autres [BroadcastChannel](#) objets configurés pour ce canal. Les messages peuvent être des objets structurés, par exemple des objets imbriqués et des tableaux.

```
broadcastChannel.close()
```

✓

Ferme l' [BroadcastChannel](#) objet, l'ouvrant à la récupération de place.

Un [BroadcastChannel](#) objet a un **nom de canal** et un **drapeau fermé** .

Les étapes du constructeur sont : **new BroadcastChannel (name)**

1. Définissez le nom de [ce canal](#) sur *name* .
2. Définissez [cet](#) indicateur [de fermeture](#) sur false.

Les **name** étapes du getter consistent à renvoyer le nom de [ce canal](#) .

Un [BroadcastChannel](#) objet est dit **éligible à la messagerie** lorsque son [objet global pertinent](#) est soit :

- un [Window](#) objet dont [l'associé Document](#) est [pleinement actif](#) , ou
- un [WorkerGlobalScope](#) objet dont le drapeau [de fermeture](#) est faux et dont le [travailleur](#) n'est pas un [travailleur suspendable](#) .

Les étapes de la méthode sont : [postMessage \(message\)](#)

1. Si [cela](#) n'est pas [éligible pour la messagerie](#) , revenez.
2. Si [ce](#) drapeau [fermé](#) est vrai, lancez un ["InvalidStateError" DOMException](#) .
3. Laissez *sérialisé* être [StructuredSerialize](#) ( *message* ). Renvoyez toutes les exceptions.
4. Soit *sourceOrigin* l' [origine](#) de [cet](#) objet [de paramètres pertinent](#) .
5. Laissez *sourceStorageKey* être le résultat de l'exécution de [l'obtention d'une clé de stockage à des fins autres que de stockage](#) avec [cet objet de paramètres](#) pertinent .
6. Soit *destinations* une liste d' [BroadcastChannel](#) objets répondant aux critères suivants :
  - Ils sont [éligibles à la messagerie](#) .
  - Le résultat de l'exécution de [l'obtention d'une clé de stockage à des fins autres que de stockage](#) avec leur [objet de paramètres pertinent est égal à sourceStorageKey](#) .
  - Leur [nom de chaîne est](#) le nom de [cette chaîne](#) .
7. Supprimer *la source* des *destinations* .
8. Trier *les destinations* de sorte que tous [BroadcastChannel](#) les objets dont [les agents pertinents](#) sont les mêmes soient triés par ordre de création, le plus ancien en premier. (Cela ne définit pas un ordre complet. Dans cette contrainte, les agents utilisateurs peuvent trier la liste de n'importe quelle manière [définie par l'implémentation](#) .)
9. Pour chaque *destination* dans *destinations* , [mettez en file d'attente une tâche globale](#) sur la [source de la tâche de manipulation DOM](#) en fonction de [l'objet global pertinent](#) de *destination* pour effectuer les étapes suivantes :
  - Si [l'indicateur fermé](#) de *destination* est vrai, alors abandonnez ces étapes.
  - Soit *targetRealm* le [domaine pertinent](#) de *la destination* .

- Laissez les données être [StructuredDeserialize](#) ( *serialized* , *targetRealm* ).  
  
Si cela lève une exception, attrapez-la, [déclenchez un événement](#) nommé `messageerror` à *destination* , en utilisant `MessageEvent`, avec l' `origin` attribut initialisé à la [sérialisation](#) de *sourceOrigin* , puis abandonnez ces étapes.
- [Déclenchez un événement](#) nommé `message` à *destination* , en utilisant `MessageEvent`, avec l' `data` attribut initialisé à *data* et l' `origin` attribut initialisé à la [sérialisation](#) de *sourceOrigin* .

Alors qu'un `BroadcastChannel` objet dont l'[indicateur fermé](#) est faux a un écouteur d'événement enregistré pour `message` ou `messageerror` événements, il doit y avoir une référence forte entre l' [objet global pertinent](#) `BroadcastChannel` de l'objet et l' objet lui-même. `BroadcastChannel`

Les `close()` étapes de la méthode consistent à définir l'[indicateur fermé](#) de `this` sur `true`.

*Les auteurs sont fortement encouragés à fermer explicitement `BroadcastChannel` les objets lorsqu'ils ne sont plus nécessaires, afin qu'ils puissent être ramassés. Créer de nombreux `BroadcastChannel` objets et les supprimer tout en les laissant avec un écouteur d'événement et sans les fermer peut conduire à une fuite de mémoire apparente, puisque les objets continueront à vivre aussi longtemps qu'ils auront un écouteur d'événement (ou jusqu'à ce que leur page ou travailleur soit fermé ).*

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par tous les objets implémentant l' `BroadcastChannel` interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<code>onmessage</code> ✓ MDN	<code>message</code>
<code>onmessageerror</code> ✓ MDN	<code>messageerror</code>

Supposons qu'une page veuille savoir quand l'utilisateur se déconnecte, même lorsqu'il le fait depuis un autre onglet du même site :

```
var authChannel = new BroadcastChannel('auth');
authChannel.onmessage = function (event) {
  if (event.data == 'logout')
```

```

    showLogout();
}

function logoutRequested() {
    // called when the user asks us to log them out
    doLogout();
    showLogout();
    authChannel.postMessage('logout');
}

function doLogout() {
    // actually log the user out (e.g. clearing cookies)
    // ...
}

function showLogout() {
    // update the UI to indicate we're logged out
    // ...
}

```

## 1. [10 travailleurs du Web](#)

### 1. [10.1 Présentation](#)

#### 1. [10.1.1 Portée](#)

#### 2. [10.1.2 Exemples](#)

1. [10.1.2.1 Un travailleur de fond qui calcule les chiffres](#)
2. [10.1.2.2 Utiliser un module JavaScript en tant que worker](#)
3. [10.1.2.3 Présentation des travailleurs partagés](#)
4. [10.1.2.4 État partagé utilisant un travailleur partagé](#)
5. [10.1.2.5 Délégation](#)
6. [10.1.2.6 Fournir des bibliothèques](#)

#### 3. [10.1.3 Tutoriels](#)

1. [10.1.3.1 Créer un worker dédié](#)
2. [10.1.3.2 Communiquer avec un intervenant dédié](#)
3. [10.1.3.3 Travailleurs partagés](#)

### 2. [10.2 Infrastructures](#)

#### 1. [10.2.1 Le périmètre mondial](#)

1. [10.2.1.1 L' `WorkerGlobalScope` interface commune](#)
2. [10.2.1.2 Travailleurs dédiés `et DedicatedWorkerGlobalScope` interface](#)

3. [10.2.1.3 Travailleurs partagés et `SharedWorkerGlobalScope` interface](#)
  2. [10.2.2 La boucle événementielle](#)
  3. [10.2.3 La durée de vie du travailleur](#)
  4. [10.2.4 Modèle de traitement](#)
  5. [10.2.5 Erreurs de script d'exécution](#)
  6. [10.2.6 Création de nœuds de calcul](#)
    1. [10.2.6.1 Le `AbstractWorker` mélange](#)
    2. [10.2.6.2 Paramètres de script pour les agents](#)
    3. [10.2.6.3 Travailleurs dédiés et `Worker` interface](#)
    4. [10.2.6.4 Travailleurs partagés et `SharedWorker` interface](#)
  7. [10.2.7 Capacités matérielles simultanées](#)
3. [10.3 API disponibles pour les travailleurs](#)
  1. [10.3.1 Importation de scripts et de bibliothèques](#)
  2. [10.3.2 L' `WorkerNavigator` interface](#)
  3. [10.3.3 L' `WorkerLocation` interface](#)

## 10 travailleurs du Web



### 10.1 Présentation

#### 10.1.1 Portée

*Cette section est non normative.*

Cette spécification définit une API pour exécuter des scripts en arrière-plan indépendamment de tout script d'interface utilisateur.

Cela permet des scripts de longue durée qui ne sont pas interrompus par des scripts qui répondent aux clics ou à d'autres interactions de l'utilisateur, et permet d'exécuter de longues tâches sans céder pour que la page reste réactive.

Les travailleurs (comme ces scripts d'arrière-plan sont appelés ici) sont relativement lourds et ne sont pas destinés à être utilisés en grand nombre. Par exemple, il serait inapproprié de lancer un ouvrier pour chaque pixel d'une image de quatre



mégapixels. Les exemples ci-dessous montrent quelques utilisations appropriées des travailleurs.

Généralement, on s'attend à ce que les travailleurs aient une longue durée de vie, aient un coût de performances de démarrage élevé et un coût de mémoire par instance élevé.

## 10.1.2 Exemples

*Cette section est non normative.*

Il existe une variété d'utilisations que les travailleurs peuvent faire. Les sous-sections suivantes montrent divers exemples de cette utilisation.

### 10.1.2.1 Un travailleur de fond qui calcule les chiffres

*Cette section est non normative.*

L'utilisation la plus simple des travailleurs consiste à effectuer une tâche coûteuse en calculs sans interrompre l'interface utilisateur.

Dans cet exemple, le document principal génère un worker pour calculer (naïvement) les nombres premiers, et affiche progressivement le nombre premier trouvé le plus récemment.

La page principale est la suivante :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Worker example: One-core computation</title>
  </head>
  <body>
    <p>The highest prime number discovered so far is: <output
id="result"></output></p>
    <script>
      var worker = new Worker('worker.js');
      worker.onmessage = function (event) {
        document.getElementById('result').textContent = event.data;
      };
    </script>
  </body>
</html>
```

```
</script>
</body>
</html>
```

L' `Worker()` appel du constructeur crée un travailleur et renvoie un `Worker` objet représentant ce travailleur, qui est utilisé pour communiquer avec le travailleur. Le gestionnaire d'événements de cet objet `onmessage` permet au code de recevoir des messages du travailleur.

Le travailleur lui-même est le suivant :

```
var n = 1;
search: while (true) {
  n += 1;
  for (var i = 2; i <= Math.sqrt(n); i += 1)
    if (n % i == 0)
      continue search;
  // found a prime!
  postMessage(n);
}
```

La majeure partie de ce code est simplement une recherche non optimisée d'un nombre premier. La `postMessage()` méthode est utilisée pour renvoyer un message à la page lorsqu'un premier est trouvé.

[Voir cet exemple en ligne](#) .

### 10.1.2.2 Utiliser un module JavaScript en tant que worker

*Cette section est non normative.*

Jusqu'à présent, tous nos exemples montrent des travailleurs qui exécutent [des scripts classiques](#) . Les travailleurs peuvent à la place être instanciés à l'aide [de scripts de module](#) , qui présentent les avantages habituels : la possibilité d'utiliser l' `import` instruction JavaScript pour importer d'autres modules ; mode strict par défaut ; et des déclarations de haut niveau ne polluant pas le périmètre global du travailleur.

Comme l' `import` instruction est disponible, la `importScripts()` méthode échouera automatiquement à l'intérieur des modules workers.

Dans cet exemple, le document principal utilise un agent pour effectuer une manipulation d'image hors thread principal. Il importe les filtres utilisés depuis un autre module.

La page principale est la suivante :

```
<!DOCTYPE html>
<html lang="en">
<meta charset="utf-8">
<title>Worker example: image decoding</title>

<p>
  <label>
    Type an image URL to decode
    <input type="url" id="image-url" list="image-list">
    <datalist id="image-list">
      <option
value="https://html.spec.whatwg.org/images/drawImage.png">
      <option
value="https://html.spec.whatwg.org/images/robots.jpeg">
      <option
value="https://html.spec.whatwg.org/images/arcTo2.png">
    </datalist>
  </label>
</p>

<p>
  <label>
    Choose a filter to apply
    <select id="filter">
      <option value="none">none</option>
      <option value="grayscale">grayscale</option>
      <option value="brighten">brighten by 20%</option>
    </select>
  </label>
</p>

<div id="output"></div>

<script type="module">
  const worker = new Worker("worker.js", { type: "module" });
  worker.onmessage = receiveFromWorker;

  const url = document.querySelector("#image-url");
  const filter = document.querySelector("#filter");
  const output = document.querySelector("#output");
```

```

url.oninput = updateImage;
filter.oninput = sendToWorker;

let imageData, context;

function updateImage() {
  const img = new Image();
  img.src = url.value;

  img.onload = () => {
    const canvas = document.createElement("canvas");
    canvas.width = img.width;
    canvas.height = img.height;

    context = canvas.getContext("2d");
    context.drawImage(img, 0, 0);
    imageData = context.getImageData(0, 0, canvas.width,
    canvas.height);

    sendToWorker();
    output.replaceChildren(canvas);
  };
}

function sendToWorker() {
  worker.postMessage({ imageData, filter: filter.value });
}

function receiveFromWorker(e) {
  context.putImageData(e.data, 0, 0);
}
</script>

```

Le fichier travailleur est alors :

```

import * as filters from "../filters.js";

self.onmessage = e => {
  const { imageData, filter } = e.data;
  filters[filter](imageData);
}

```

```
self.postMessage(imageData, [imageData.data.buffer]);  
};
```

Qui importe le fichier `filters.js`:

```
export function none() {}  
  
export function grayscale({ data: d }) {  
  for (let i = 0; i < d.length; i += 4) {  
    const [r, g, b] = [d[i], d[i + 1], d[i + 2]];  
  
    // CIE luminance for the RGB  
    // The human eye is bad at seeing red and blue, so we de-  
    emphasize them.  
    d[i] = d[i + 1] = d[i + 2] = 0.2126 * r + 0.7152 * g + 0.0722 *  
b;  
  }  
};  
  
export function brighten({ data: d }) {  
  for (let i = 0; i < d.length; ++i) {  
    d[i] *= 1.2;  
  }  
};
```

[Voir cet exemple en ligne](#) .

### 10.1.2.3 Présentation des travailleurs partagés



*Cette section est non normative.*

Cette section présente les nœuds de calcul partagés à l'aide d'un exemple Hello World. Les nœuds de calcul partagés utilisent des API légèrement différentes, car chaque nœud de calcul peut avoir plusieurs connexions.

Ce premier exemple montre comment vous vous connectez à un travailleur et comment un travailleur peut renvoyer un message à la page lorsqu'il s'y connecte. Les messages reçus sont affichés dans un journal.

Voici la page HTML :

```

<!DOCTYPE HTML>
<html lang="en">
<meta charset="utf-8">
<title>Shared workers: demo 1</title>
<pre id="log">Log:</pre>
<script>
    var worker = new SharedWorker('test.js');
    var log = document.getElementById('log');
    worker.port.onmessage = function(e) { // note: not
worker.onmessage!
        log.textContent += '\n' + e.data;
    }
</script>

```

Voici le worker JavaScript :

```

onconnect = function(e) {
    var port = e.ports[0];
    port.postMessage('Hello World!');
}

```

[Voir cet exemple en ligne](#) .

---

Ce deuxième exemple étend le premier en modifiant deux choses : premièrement, les messages sont reçus en utilisant `addEventListener()` au lieu d'un [attribut IDL de gestionnaire d'événements](#) , et deuxièmement, un message est envoyé *au* travailleur, obligeant le travailleur à envoyer un autre message en retour. Les messages reçus sont à nouveau affichés dans un journal.

Voici la page HTML :

```

<!DOCTYPE HTML>
<html lang="en">
<meta charset="utf-8">
<title>Shared workers: demo 2</title>
<pre id="log">Log:</pre>
<script>
    var worker = new SharedWorker('test.js');
    var log = document.getElementById('log');
    worker.port.addEventListener('message', function(e) {

```

```

    log.textContent += '\n' + e.data;
  }, false);

  worker.port.start(); // note: need this when using
  addEventListener

  worker.port.postMessage('ping');
</script>

```

Voici le worker JavaScript :

```

onconnect = function(e) {
  var port = e.ports[0];
  port.postMessage('Hello World!');
  port.onmessage = function(e) {
    port.postMessage('pong'); // not e.ports[0].postMessage!
    // e.target.postMessage('pong'); would work also
  }
}

```

[Voir cet exemple en ligne](#) .

---

Enfin, l'exemple est étendu pour montrer comment deux pages peuvent se connecter au même worker ; dans ce cas, la deuxième page est simplement dans un [iframe](#) sur la première page, mais le même principe s'appliquerait à une page entièrement séparée dans un [traversable de niveau supérieur](#) séparé .

Voici la page HTML externe :

```

<!DOCTYPE HTML>
<html lang="en">
<meta charset="utf-8">
<title>Shared workers: demo 3</title>
<pre id="log">Log:</pre>
<script>
  var worker = new SharedWorker('test.js');
  var log = document.getElementById('log');
  worker.port.addEventListener('message', function(e) {
    log.textContent += '\n' + e.data;
  }, false);
  worker.port.start();

```

```

    worker.port.postMessage('ping');
</script>
<iframe src="inner.html"></iframe>

```

Voici la page HTML interne :

```

<!DOCTYPE HTML>
<html lang="en">
<meta charset="utf-8">
<title>Shared workers: demo 3 inner frame</title>
<pre id=log>Inner log:</pre>
<script>
    var worker = new SharedWorker('test.js');
    var log = document.getElementById('log');
    worker.port.onmessage = function(e) {
        log.textContent += '\n' + e.data;
    }
</script>

```

Voici le worker JavaScript :

```

var count = 0;
onconnect = function(e) {
    count += 1;
    var port = e.ports[0];
    port.postMessage('Hello World! You are connection #' + count);
    port.onmessage = function(e) {
        port.postMessage('pong');
    }
}

```

[Voir cet exemple en ligne](#) .

#### 10.1.2.4 État partagé utilisant un travailleur partagé

*Cette section est non normative.*

Dans cet exemple, plusieurs fenêtres (visionneuses) peuvent être ouvertes qui affichent toutes la même carte. Toutes les fenêtres partagent les mêmes informations cartographiques, avec un seul travailleur coordonnant tous les visualiseurs. Chaque spectateur peut se déplacer indépendamment, mais s'il définit des données sur la carte, tous les spectateurs sont mis à jour.



La page principale n'est pas intéressante, elle fournit simplement un moyen d'ouvrir les visualiseurs :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Workers example: Multiviewer</title>
    <script>
      function openViewer() {
        window.open('viewer.html');
      }
    </script>
  </head>
  <body>
    <p><button type=button onclick="openViewer()">Open a new
      viewer</button></p>
    <p>Each viewer opens in a new window. You can have as many
      viewers
      as you like, they all view the same data.</p>
  </body>
</html>
```

Le spectateur est plus impliqué :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Workers example: Multiviewer viewer</title>
    <script>
      var worker = new SharedWorker('worker.js', 'core');

      // CONFIGURATION
      function configure(event) {
        if (event.data.substr(0, 4) != 'cfg ') return;
        var name = event.data.substr(4).split(' ', 1)[0];
        // update display to mention our name is name
        document.getElementsByTagName('h1')[0].textContent += ' ' +
name;
        // no longer need this listener
        worker.port.removeEventListener('message', configure, false);
```

```

    }

    worker.port.addEventListener('message', configure, false);

    // MAP
    function paintMap(event) {
        if (event.data.substr(0, 4) != 'map ') return;
        var data = event.data.substr(4).split(',');
        // display tiles data[0] .. data[8]
        var canvas = document.getElementById('map');
        var context = canvas.getContext('2d');
        for (var y = 0; y < 3; y += 1) {
            for (var x = 0; x < 3; x += 1) {
                var tile = data[y * 3 + x];
                if (tile == '0')
                    context.fillStyle = 'green';
                else
                    context.fillStyle = 'maroon';
                context.fillRect(x * 50, y * 50, 50, 50);
            }
        }
    }

    worker.port.addEventListener('message', paintMap, false);

    // PUBLIC CHAT
    function updatePublicChat(event) {
        if (event.data.substr(0, 4) != 'txt ') return;
        var name = event.data.substr(4).split(' ', 1)[0];
        var message = event.data.substr(4 + name.length + 1);
        // display "<name> message" in public chat
        var public = document.getElementById('public');
        var p = document.createElement('p');
        var n = document.createElement('button');
        n.textContent = '<' + name + '> ';
        n.onclick = function () { worker.port.postMessage('msg ' +
name); };
        p.appendChild(n);
        var m = document.createElement('span');
        m.textContent = message;
        p.appendChild(m);
        public.appendChild(p);
    }

```

```
worker.port.addEventListener('message', updatePublicChat,  
false);
```

```
// PRIVATE CHAT
```

```
function startPrivateChat(event) {
```

```
    if (event.data.substr(0, 4) != 'msg ') return;
```

```
    var name = event.data.substr(4).split(' ', 1)[0];
```

```
    var port = event.ports[0];
```

```
    // display a private chat UI
```

```
    var ul = document.getElementById('private');
```

```
    var li = document.createElement('li');
```

```
    var h3 = document.createElement('h3');
```

```
    h3.textContent = 'Private chat with ' + name;
```

```
    li.appendChild(h3);
```

```
    var div = document.createElement('div');
```

```
    var addMessage = function(name, message) {
```

```
        var p = document.createElement('p');
```

```
        var n = document.createElement('strong');
```

```
        n.textContent = '<' + name + '> ';
```

```
        p.appendChild(n);
```

```
        var t = document.createElement('span');
```

```
        t.textContent = message;
```

```
        p.appendChild(t);
```

```
        div.appendChild(p);
```

```
    };
```

```
    port.onmessage = function (event) {
```

```
        addMessage(name, event.data);
```

```
    };
```

```
    li.appendChild(div);
```

```
    var form = document.createElement('form');
```

```
    var p = document.createElement('p');
```

```
    var input = document.createElement('input');
```

```
    input.size = 50;
```

```
    p.appendChild(input);
```

```
    p.appendChild(document.createTextNode(' '));
```

```
    var button = document.createElement('button');
```

```
    button.textContent = 'Post';
```

```
    p.appendChild(button);
```

```
    form.onsubmit = function () {
```

```
        port.postMessage(input.value);
```

```
        addMessage('me', input.value);
```

```

        input.value = '';
        return false;
    };
    form.appendChild(p);
    li.appendChild(form);
    ul.appendChild(li);
}

worker.port.addEventListener('message', startPrivateChat,
false);

worker.port.start();
</script>
</head>
<body>
    <h1>Viewer</h1>
    <h2>Map</h2>
    <p><canvas id="map" height=150 width=150></canvas></p>
    <p>
        <button type=button onclick="worker.port.postMessage('mov
left') ">>Left</button>
        <button type=button onclick="worker.port.postMessage('mov
up') ">>Up</button>
        <button type=button onclick="worker.port.postMessage('mov
down') ">>Down</button>
        <button type=button onclick="worker.port.postMessage('mov
right') ">>Right</button>
        <button type=button onclick="worker.port.postMessage('set
0') ">>Set 0</button>
        <button type=button onclick="worker.port.postMessage('set
1') ">>Set 1</button>
    </p>
    <h2>Public Chat</h2>
    <div id="public"></div>
    <form onsubmit="worker.port.postMessage('txt ' + message.value);
message.value = ''; return false;">
    <p>
        <input type="text" name="message" size="50">
        <button>Post</button>
    </p>
</form>
    <h2>Private Chat</h2>
    <ul id="private"></ul>
</body>

```

```
</html>
```

Il y a plusieurs éléments clés à noter sur la façon dont le visualiseur est écrit.

**Plusieurs auditeurs** . Au lieu d'une fonction de traitement de message unique, le code attache ici plusieurs écouteurs d'événement, chacun effectuant une vérification rapide pour voir s'il est pertinent pour le message. Dans cet exemple, cela ne fait pas beaucoup de différence, mais si plusieurs auteurs voulaient collaborer en utilisant un seul port pour communiquer avec un travailleur, cela permettrait un code indépendant au lieu de devoir tous apporter des modifications à une seule fonction de gestion des événements.

L'enregistrement des écouteurs d'événements de cette manière vous permet également de désenregistrer des écouteurs spécifiques lorsque vous en avez terminé avec eux, comme c'est le cas avec la `configure()` méthode de cet exemple.

Enfin, le travailleur :

```
var nextName = 0;

function getNextName() {
    // this could use more friendly names
    // but for now just return a number
    return nextName++;
}

var map = [
    [0, 0, 0, 0, 0, 0, 0],
    [1, 1, 0, 1, 0, 1, 1],
    [0, 1, 0, 1, 0, 0, 0],
    [0, 1, 0, 1, 0, 1, 1],
    [0, 0, 0, 1, 0, 0, 0],
    [1, 0, 0, 1, 1, 1, 1],
    [1, 1, 0, 1, 1, 0, 1],
];

function wrapX(x) {
    if (x < 0) return wrapX(x + map[0].length);
    if (x >= map[0].length) return wrapX(x - map[0].length);
    return x;
}

function wrapY(y) {
    if (y < 0) return wrapY(y + map.length);
    if (y >= map[0].length) return wrapY(y - map.length);
}
```

```

    return y;
}

function wrap(val, min, max) {
    if (val < min)
        return val + (max-min)+1;
    if (val > max)
        return val - (max-min)-1;
    return val;
}

function sendMapData(viewer) {
    var data = '';
    for (var y = viewer.y-1; y <= viewer.y+1; y += 1) {
        for (var x = viewer.x-1; x <= viewer.x+1; x += 1) {
            if (data != '')
                data += ',';
            data += map[wrap(y, 0, map[0].length-1)][wrap(x, 0,
map.length-1)];
        }
    }
    viewer.port.postMessage('map ' + data);
}

var viewers = {};
onconnect = function (event) {
    var name = getNextName();
    event.ports[0]._data = { port: event.ports[0], name: name, x: 0,
y: 0, };
    viewers[name] = event.ports[0]._data;
    event.ports[0].postMessage('cfg ' + name);
    event.ports[0].onmessage = getMessage;
    sendMapData(event.ports[0]._data);
};

function getMessage(event) {
    switch (event.data.substr(0, 4)) {
        case 'mov ':
            var direction = event.data.substr(4);
            var dx = 0;
            var dy = 0;

```

```

        switch (direction) {
            case 'up': dy = -1; break;
            case 'down': dy = 1; break;
            case 'left': dx = -1; break;
            case 'right': dx = 1; break;
        }

        event.target._data.x = wrapX(event.target._data.x + dx);
        event.target._data.y = wrapY(event.target._data.y + dy);
        sendMapData(event.target._data);
        break;
    case 'set ':
        var value = event.data.substr(4);
        map[event.target._data.y][event.target._data.x] = value;
        for (var viewer in viewers)
            sendMapData(viewers[viewer]);
        break;
    case 'txt ':
        var name = event.target._data.name;
        var message = event.data.substr(4);
        for (var viewer in viewers)
            viewers[viewer].port.postMessage('txt ' + name + ' ' +
            message);
        break;
    case 'msg ':
        var party1 = event.target._data;
        var party2 = viewers[event.data.substr(4).split(' ', 1)[0]];
        if (party2) {
            var channel = new MessageChannel();
            party1.port.postMessage('msg ' + party2.name,
            [channel.port1]);
            party2.port.postMessage('msg ' + party1.name,
            [channel.port2]);
        }
        break;
    }
}

```

**Connexion à plusieurs pages** . Le script utilise l' [onconnect](#) écouteur d'événement pour écouter plusieurs connexions.

**Chaînes directes** . Lorsque le travailleur reçoit un message "msg" d'un spectateur nommant un autre spectateur, il établit une connexion directe entre les deux, de

sorte que les deux spectateurs peuvent communiquer directement sans que le travailleur n'ait à proxy tous les messages.

[Voir cet exemple en ligne](#) .

#### 10.1.2.5 Délégation

*Cette section est non normative.*

Avec la généralisation des processeurs multicœurs, une façon d'obtenir de meilleures performances consiste à répartir les tâches coûteuses en calcul entre plusieurs travailleurs. Dans cet exemple, une tâche coûteuse en calcul qui doit être effectuée pour chaque nombre de 1 à 10 000 000 est confiée à dix sous-travailleurs.

La page principale est la suivante, elle rapporte juste le résultat :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Worker example: Multicore computation</title>
  </head>
  <body>
    <p>Result: <output id="result"></output></p>
    <script>
      var worker = new Worker('worker.js');
      worker.onmessage = function (event) {
        document.getElementById('result').textContent = event.data;
      };
    </script>
  </body>
</html>
```

Le travailleur lui-même est le suivant :

```
// settings
var num_workers = 10;
var items_per_worker = 1000000;

// start the workers
var result = 0;
var pending_workers = num_workers;
```



```

for (var i = 0; i < num_workers; i += 1) {
    var worker = new Worker('core.js');
    worker.postMessage(i * items_per_worker);
    worker.postMessage((i+1) * items_per_worker);
    worker.onmessage = storeResult;
}

// handle the results
function storeResult(event) {
    result += 1*event.data;
    pending_workers -= 1;
    if (pending_workers <= 0)
        postMessage(result); // finished!
}

```

Il se compose d'une boucle pour démarrer les sous-travailleurs, puis d'un gestionnaire qui attend que tous les sous-travailleurs répondent.

Les sous-travailleurs sont implémentés comme suit :

```

var start;
onmessage = getStart;
function getStart(event) {
    start = 1*event.data;
    onmessage = getEnd;
}

var end;
function getEnd(event) {
    end = 1*event.data;
    onmessage = null;
    work();
}

function work() {
    var result = 0;
    for (var i = start; i < end; i += 1) {
        // perform some complex calculation here
        result += 1;
    }
    postMessage(result);
    close();
}

```

```
}
```

Ils reçoivent deux nombres en deux événements, effectuent le calcul pour la plage de nombres ainsi spécifiée, puis rapportent le résultat au parent.

[Voir cet exemple en ligne](#) .

#### 10.1.2.6 Fournir des bibliothèques

*Cette section est non normative.*

Supposons qu'une bibliothèque de cryptographie soit mise à disposition et fournisse trois tâches :

##### **Générer une paire de clés publique/privée**

Prend un port, sur lequel il enverra deux messages, d'abord la clé publique puis la clé privée.

##### **Étant donné un texte en clair et une clé publique, renvoie le texte chiffré correspondant**

Prend un port, auquel n'importe quel nombre de messages peut être envoyé, le premier donnant la clé publique, et le reste donnant le texte en clair, chacun étant crypté puis envoyé sur ce même canal que le texte chiffré. L'utilisateur peut fermer le port lorsqu'il a fini de chiffrer le contenu.

##### **Étant donné un texte chiffré et une clé privée, renvoie le texte en clair correspondant**

Prend un port, auquel n'importe quel nombre de messages peut être envoyé, le premier donnant la clé privée, et le reste donnant le texte chiffré, chacun étant déchiffré puis envoyé sur ce même canal que le texte en clair. L'utilisateur peut fermer le port lorsqu'il a fini de déchiffrer le contenu.

La bibliothèque elle-même est la suivante :

```
function handleMessage(e) {
  if (e.data == "genkeys")
    genkeys(e.ports[0]);
  else if (e.data == "encrypt")
    encrypt(e.ports[0]);
  else if (e.data == "decrypt")
    decrypt(e.ports[0]);
}

function genkeys(p) {
  var keys = _generateKeyPair();
```

```
p.postMessage(keys[0]);  
p.postMessage(keys[1]);  
}
```

```
function encrypt(p) {  
  var key, state = 0;  
  p.onmessage = function (e) {  
    if (state == 0) {  
      key = e.data;  
      state = 1;  
    } else {  
      p.postMessage(_encrypt(key, e.data));  
    }  
  };  
}
```

```
function decrypt(p) {  
  var key, state = 0;  
  p.onmessage = function (e) {  
    if (state == 0) {  
      key = e.data;  
      state = 1;  
    } else {  
      p.postMessage(_decrypt(key, e.data));  
    }  
  };  
}
```

```
// support being used as a shared worker as well as a dedicated  
worker
```

```
if ('onmessage' in this) // dedicated worker  
  onmessage = handleMessage;  
else // shared worker  
  onconnect = function (e) { e.port.onmessage = handleMessage; }
```

```
// the "crypto" functions:
```

```
function _generateKeyPair() {  
  return [Math.random(), Math.random()];  
}
```

```
function _encrypt(k, s) {
  return 'encrypted-' + k + ' ' + s;
}
```

```
function _decrypt(k, s) {
  return s.substr(s.indexOf(' ') + 1);
}
```

Notez que les fonctions de chiffrement ici ne sont que des stubs et ne font pas de vraie cryptographie.

Cette bibliothèque pourrait être utilisée comme suit :

```
<!DOCTYPE HTML>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <title>Worker example: Crypto library</title>
    <script>
      const cryptoLib = new Worker('libcrypto-v1.js'); // or could use
      'libcrypto-v2.js'

      function startConversation(source, message) {
        const messageChannel = new MessageChannel();
        source.postMessage(message, [messageChannel.port2]);
        return messageChannel.port1;
      }

      function getKeys() {
        let state = 0;

        startConversation(cryptoLib, "genkeys").onmessage = function
        (e) {
          if (state === 0)
            document.getElementById('public').value = e.data;
          else if (state === 1)
            document.getElementById('private').value = e.data;
          state += 1;
        };
      }

      function enc() {
        const port = startConversation(cryptoLib, "encrypt");
        port.postMessage(document.getElementById('public').value);
        port.postMessage(document.getElementById('input').value);
      }
    </script>
  </head>
  <body>
    <div>
      <div>
        <input type="text" value="Public key" />
      </div>
      <div>
        <input type="text" value="Private key" />
      </div>
      <div>
        <input type="text" value="Message to encrypt" />
      </div>
      <div>
        <input type="button" value="Encrypt" />
      </div>
    </div>
  </body>
</html>
```

```

    port.onmessage = function (e) {
        document.getElementById('input').value = e.data;
        port.close();
    };
}

function dec() {
    const port = startConversation(cryptoLib, "decrypt");
    port.postMessage(document.getElementById('private').value);
    port.postMessage(document.getElementById('input').value);
    port.onmessage = function (e) {
        document.getElementById('input').value = e.data;
        port.close();
    };
}

</script>
<style>
    textarea { display: block; }
</style>
</head>
<body onload="getKeys()">
    <fieldset>
        <legend>Keys</legend>
        <p><label>Public Key: <textarea
id="public"></textarea></label></p>
        <p><label>Private Key: <textarea
id="private"></textarea></label></p>
    </fieldset>
    <p><label>Input: <textarea id="input"></textarea></label></p>
    <p><button onclick="enc()">Encrypt</button> <button
onclick="dec()">Decrypt</button></p>
</body>
</html>

```

Une version ultérieure de l'API, cependant, pourrait vouloir décharger tout le travail de chiffrement sur les sous-travailleurs. Cela pourrait être fait comme suit :

```

function handleMessage(e) {
    if (e.data == "genkeys")
        genkeys(e.ports[0]);
    else if (e.data == "encrypt")
        encrypt(e.ports[0]);
    else if (e.data == "decrypt")

```

```

    decrypt(e.ports[0]);
}

function genkeys(p) {
    var generator = new Worker('libcrypto-v2-generator.js');
    generator.postMessage('', [p]);
}

function encrypt(p) {
    p.onmessage = function (e) {
        var key = e.data;
        var encryptor = new Worker('libcrypto-v2-encryptor.js');
        encryptor.postMessage(key, [p]);
    };
}

function encrypt(p) {
    p.onmessage = function (e) {
        var key = e.data;
        var decryptor = new Worker('libcrypto-v2-decryptor.js');
        decryptor.postMessage(key, [p]);
    };
}

// support being used as a shared worker as well as a dedicated
worker
if ('onmessage' in this) // dedicated worker
    onmessage = handleMessage;
else // shared worker
    onconnect = function (e) { e.ports[0].onmessage = handleMessage
};

```

Les petits sous-travailleurs seraient alors les suivants.

Pour générer des paires de clés :

```

onmessage = function (e) {
    var k = _generateKeyPair();
    e.ports[0].postMessage(k[0]);
    e.ports[0].postMessage(k[1]);
    close();
}

```

```
function _generateKeyPair() {
    return [Math.random(), Math.random()];
}
```

Pour chiffrer :

```
onmessage = function (e) {
    var key = e.data;
    e.ports[0].onmessage = function (e) {
        var s = e.data;
        postMessage(_encrypt(key, s));
    }
}

function _encrypt(k, s) {
    return 'encrypted-' + k + ' ' + s;
}
```

Pour décrypter :

```
onmessage = function (e) {
    var key = e.data;
    e.ports[0].onmessage = function (e) {
        var s = e.data;
        postMessage(_decrypt(key, s));
    }
}

function _decrypt(k, s) {
    return s.substr(s.indexOf(' ')+1);
}
```

Remarquez que les utilisateurs de l'API n'ont même pas besoin de savoir que cela se produit — l'API n'a pas changé ; la bibliothèque peut déléguer à des sous-travailleurs sans changer son API, même si elle accepte des données à l'aide de canaux de messages.

[Voir cet exemple en ligne](#) .

### 10.1.3 Tutoriels

### 10.1.3.1 Créer un worker dédié

*Cette section est non normative.*

La création d'un travailleur nécessite une URL vers un fichier JavaScript. Le `Worker()` constructeur est invoqué avec l'URL de ce fichier comme seul argument ; un worker est alors créé et retourné :

```
var worker = new Worker('helper.js');
```

Si vous souhaitez que votre script de travail soit interprété comme un [script de module](#) au lieu du [script classique](#) par défaut , vous devez utiliser une signature légèrement différente :

```
var worker = new Worker('helper.mjs', { type: "module" });
```

### 10.1.3.2 Communiquer avec un intervenant dédié

*Cette section est non normative.*

Des travailleurs dédiés utilisent [MessagePort](#) des objets dans les coulisses et prennent ainsi en charge toutes les mêmes fonctionnalités, telles que l'envoi de données structurées, le transfert de données binaires et le transfert d'autres ports.

Pour recevoir des messages d'un agent dédié, utilisez l' [onmessage](#) [attribut IDL du gestionnaire d'événements](#) sur l' [Worker](#) objet :

```
worker.onmessage = function (event) { ... };
```

Vous pouvez également utiliser la [addEventListener\(\)](#) méthode.

*L'implicite [MessagePort](#) utilisé par les travailleurs dédiés a sa [file d'attente de messages de port](#) implicitement activée lors de sa création, il n'y a donc pas d'équivalent à la méthode [MessagePort](#) de l'interface [start\(\)](#) sur l' [Worker](#) interface.*

Pour envoyer des données à un travailleur, utilisez la [postMessage\(\)](#) méthode. Des données structurées peuvent être envoyées sur ce canal de communication. Pour envoyer [ArrayBuffer](#) des objets efficacement (en les transférant plutôt qu'en les clonant), répertoriez-les dans un tableau dans le deuxième argument.

```
worker.postMessage({
  operation: 'find-edges',
  input: buffer, // an ArrayBuffer object
  threshold: 0.6,
}, [buffer]);
```



Pour recevoir un message à l'intérieur du travailleur, l' [onmessage](#) [attribut IDL du gestionnaire d'événements](#) est utilisé.

```
onmessage = function (event) { ... };
```

Vous pouvez à nouveau utiliser la [addEventListener\(\)](#) méthode.

Dans les deux cas, les données sont fournies dans l' [data](#) attribut de l'objet événement.

Pour renvoyer des messages, vous utilisez à nouveau [postMessage\(\)](#). Il prend en charge les données structurées de la même manière.

```
postMessage(event.data.input, [event.data.input]); // transfer the  
buffer back
```

### 10.1.3.3 Travailleurs partagés



*Cette section est non normative.*

Les travailleurs partagés sont identifiés par l'URL du script utilisé pour le créer, éventuellement avec un nom explicite. Le nom permet de démarrer plusieurs instances d'un travailleur partagé particulier.

Les travailleurs partagés sont classés par [origine](#). Deux sites différents utilisant les mêmes noms n'entreront pas en collision. Cependant, si une page essaie d'utiliser le même nom de travailleur partagé qu'une autre page sur le même site, mais avec une URL de script différente, elle échouera.

La création de travailleurs partagés se fait à l'aide du [SharedWorker\(\)](#) constructeur. Ce constructeur prend l'URL du script à utiliser pour son premier argument et le nom du travailleur, le cas échéant, comme deuxième argument.

```
var worker = new SharedWorker('service.js');
```

La communication avec les travailleurs partagés se fait avec [MessagePort](#) des objets explicites. L'objet renvoyé par le [SharedWorker\(\)](#) constructeur contient une référence au port sur son [port](#) attribut.

```
worker.port.onmessage = function (event) { ... };  
worker.port.postMessage('some message');  
worker.port.postMessage({ foo: 'structured', bar: ['data', 'also',  
'possible']});
```

À l'intérieur du travailleur partagé, les nouveaux clients du travailleur sont annoncés à l'aide de l' [connect](#) événement. Le port du nouveau client est donné par l' [source](#) attribut de l'objet événement.

```
onconnect = function (event) {  
  var newPort = event.source;  
  // set up a listener  
  newPort.onmessage = function (event) { ... };  
  // send a message back to the port  
  newPort.postMessage('ready!'); // can also send structured data,  
  of course  
};
```

## 10.2 Infrastructures

Cette norme définit deux types de travailleurs : les travailleurs dédiés et les travailleurs partagés. Les travailleurs dédiés, une fois créés, sont liés à leur créateur, mais les ports de messages peuvent être utilisés pour communiquer depuis un travailleur dédié vers plusieurs autres contextes de navigation ou travailleurs. Les travailleurs partagés, en revanche, sont nommés et, une fois créés, tout script exécuté dans la même [origine](#) peut obtenir une référence à ce travailleur et communiquer avec lui. *Service Workers* définit un troisième type. [\[SW\]](#)

### 10.2.1 Le périmètre mondial

La portée globale est "l'intérieur" d'un travailleur.

#### 10.2.1.1 L' [WorkerGlobalScope](#) interface commune



[Exposed=Worker]

```
interface WorkerGlobalScope : EventTarget {
```

```
  readonly attribute WorkerGlobalScope self;
```

```
  readonly attribute WorkerLocation location;
```

```
readonly attribute WorkerNavigator navigator;
```

```
undefined importScripts (USVString... urls);
```

```
attribute OnErrorEventHandler onerror;
```

```
attribute EventHandler onlanguagechange;
```

```
attribute EventHandler onoffline;
```

```
attribute EventHandler ononline;
```

```
attribute EventHandler onrejectionhandled;
```

```
attribute EventHandler onunhandledrejection;
```

```
};
```

[WorkerGlobalScope](#) sert de classe de base pour des types spécifiques d'objets de portée globale de travail, notamment [DedicatedWorkerGlobalScope](#), [SharedWorkerGlobalScope](#) et [ServiceWorkerGlobalScope](#).

Un [WorkerGlobalScope](#) objet est associé à un **ensemble de propriétaires** (un [ensemble](#) d'objets [Document](#) et [WorkerGlobalScope](#)). Il est initialement vide et rempli lorsque le travailleur est créé ou obtenu.

*C'est un [ensemble](#), au lieu d'un propriétaire unique, pour accueillir [SharedWorkerGlobalScope](#) des objets.*

Un [WorkerGlobalScope](#) objet a un **type** associé ("classic" ou "module"). Il est défini lors de la création.

Un [WorkerGlobalScope](#) objet a une **url** associée (null ou une [URL](#)). Il est initialement nul.

Un [WorkerGlobalScope](#) objet a un **nom** associé (une chaîne). Il est défini lors de la création.

*Le [nom](#) peut avoir une sémantique différente pour chaque sous-classe de [WorkerGlobalScope](#). Pour [DedicatedWorkerGlobalScope](#) les instances, il s'agit simplement d'un nom fourni par le développeur, utile principalement à des fins de débogage. Pour [SharedWorkerGlobalScope](#) les instances, cela permet d'obtenir une référence à un worker partagé commun via*

le `SharedWorker()` constructeur. Pour `ServiceWorkerGlobalScope` les objets, cela n'a pas de sens (et en tant que tel n'est pas du tout exposé via l'API JavaScript).

Un `WorkerGlobalScope` objet a un **conteneur de stratégie** associé (un [conteneur de stratégie](#)). Il s'agit initialement d'un nouveau [conteneur de stratégie](#).

Un objet a une **stratégie d'intégration** `WorkerGlobalScope` associée (une [stratégie d'intégration](#)).

Un `WorkerGlobalScope` objet a une **carte de module** associée. Il s'agit d'une [carte de module](#), initialement vide.

Un objet a un booléen **de capacité isolé d'origine croisée** `WorkerGlobalScope` associé. C'est d'abord faux.

`workerGlobal.self`

✓

Renvoie `workerGlobal`.

`workerGlobal.location`

✓

Renvoie l'objet de `workerGlobal` `WorkerLocation`.

`workerGlobal.navigator`

✓

Renvoie l'objet de `workerGlobal` `WorkerNavigator`.

`workerGlobal.importScripts(...urls)`

✓

Récupère chaque [URL](#) dans `urls`, les exécute une par une dans l'ordre dans lequel elles sont transmises, puis renvoie (ou lance si quelque chose ne va pas).

L' `self` attribut doit renvoyer l' `WorkerGlobalScope` objet lui-même.

L' `location` attribut doit renvoyer l' `WorkerLocation` objet dont `WorkerGlobalScope` l'objet associé est l' `WorkerGlobalScope` objet.

*Bien que l' `WorkerLocation` objet soit créé après l' `WorkerGlobalScope` objet, ce n'est pas problématique car il ne peut pas être observé à partir du script.*

---

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par les objets implémentant l' [WorkerGlobalScope](#) interface :

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
<b>onerror</b> ✓ MDN	<a href="#">error</a>
<b>onlanguagechange</b> ✓ MDN	<a href="#">languagechange</a>
<b>onoffline</b> MDN	<a href="#">offline</a>
<b>ononline</b> MDN	<a href="#">online</a>
<b>onrejectionhandled</b>	<a href="#">rejectionhandled</a>
<b>onunhandledrejection</b>	<a href="#">unhandledrejection</a>

### 10.2.1.2 Travailleurs dédiés et [DedicatedWorkerGlobalScope](#) interface

✓ MDN

```
[Global=(Worker,DedicatedWorker),Exposed=DedicatedWorker]
```

```
interface DedicatedWorkerGlobalScope : WorkerGlobalScope {
```

```
  [Replaceable] readonly attribute DOMString name;
```

```
  undefined postMessage(any message, sequence<object> transfer);
```

```
  undefined postMessage(any message, optional
```

```
    StructuredSerializeOptions options = {});
```

```
  undefined close();
```

```
  attribute EventHandler onmessage;
```

```
  attribute EventHandler onmessageerror;
```



[DedicatedWorkerGlobalScope](#) les objets agissent comme s'ils avaient un implicite [MessagePort](#) qui leur était associé. Ce port fait partie d'un canal configuré lors de la création du nœud de calcul, mais il n'est pas exposé. Cet objet ne doit jamais être ramassé avant l' [DedicatedWorkerGlobalScope](#) objet.

Tous les messages reçus par ce port doivent être immédiatement redirigés vers l' [DedicatedWorkerGlobalScope](#) objet.

```
dedicatedWorkerGlobal.name
```



Renvoie le [nom](#) [dedicatedWorkerGlobal](#) , c'est-à-dire la valeur donnée au constructeur. Principalement utile pour le débogage. [Worker](#)

```
dedicatedWorkerGlobal.postMessage\(message \[, transfer \]\)
```



```
dedicatedWorkerGlobal.postMessage\(message \[, { transfer } \]\)
```

Clone le message et le transmet à l' [Worker](#) objet associé à [distributedWorkerGlobal](#) . [transfer](#) peut être passé sous la forme d'une liste d'objets qui doivent être transférés plutôt que clonés.

```
dedicatedWorkerGlobal.close\(\)
```



Abandonne [dedicatedWorkerGlobal](#) . [\\_](#)

Les étapes [du](#) [name](#) getter consistent à retourner [ce](#) nom . Sa valeur représente le nom donné au travailleur utilisant le constructeur, utilisé principalement à des fins de débogage. [Worker](#)

Les méthodes et sur les objets agissent comme si, lorsqu'elles étaient invoquées, elles invoquaient immédiatement le et respectif sur le port, avec les mêmes arguments, et renvoyaient la même valeur de

retour. [postMessage\(message, transfer\)](#) [postMessage\(message, options\)](#) [DedicatedWorkerGlobalScope.postMessage\(message, transfer\)](#) [postMessage\(message, options\)](#)

Pour **fermer un worker** , étant donné un *workerGlobal* , exécutez ces étapes :

1. Ignorer toutes les [tâches](#) qui ont été ajoutées aux [files d'attente de tâches](#) de [la boucle d'événements](#) de l' [agent concerné de](#) *workerGlobal* .
2. Définissez l'indicateur [de fermeture](#) de *workerGlobal* sur true. (Cela empêche toute autre tâche d'être mise en file d'attente.)

Les [close\(\)](#) étapes de la méthode consistent à [fermer un travailleur](#) étant donné [ceci](#) .

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par les objets implémentant l' [DedicatedWorkerGlobalScope](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<a href="#">onmessage</a> ✓ <a href="#">MDN</a>	<a href="#">message</a>
<a href="#">onmessageerror</a> <a href="#">MDN</a>	<a href="#">messageerror</a>

### 10.2.1.3 Travailleurs partagés et [SharedWorkerGlobalScope](#) interface

✓ [MDN](#)

```
[Global=(Worker,SharedWorker),Exposed=SharedWorker]
```

```
interface SharedWorkerGlobalScope : WorkerGlobalScope {
```

```
  [Replaceable] readonly attribute DOMString name;
```

```
  undefined close();
```

```
  attribute EventHandler onconnect;
```

```
};
```

Un [SharedWorkerGlobalScope](#) objet a une **origine de constructeur** associée , une **URL de constructeur** et des **informations d'identification** . Ils sont initialisés lors de [SharedWorkerGlobalScope](#) la création de l'objet, lors de l' [exécution d'un algorithme de travail](#) .

Les nœuds de calcul partagés reçoivent des ports de messages via [connect](#) des événements sur leur [SharedWorkerGlobalScope](#) objet pour chaque connexion.

[sharedWorkerGlobal.name](#)

✓ [MDN](#)

Renvoie le [nom](#) de *sharedWorkerGlobal* , c'est-à-dire la valeur donnée au constructeur. Plusieurs objets peuvent correspondre au même worker partagé (et ), en réutilisant le même [nom.SharedWorkerSharedWorkerSharedWorkerGlobalScope](#)

*sharedWorkerGlobal*.[close\(\)](#)



Abandonne *sharedWorkerGlobal* .

Les étapes [du](#)[name](#) getter consistent à retourner [ce](#) nom . Sa valeur représente le nom qui peut être utilisé pour obtenir une référence au travailleur utilisant le constructeur.[SharedWorker](#)

Les [close\(\)](#) étapes de la méthode consistent à [fermer un travailleur](#) étant donné [ceci](#) .

---

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par les objets implémentant l' [SharedWorkerGlobalScope](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<a href="#">onconnect</a> ✓	<a href="#">connect</a>

## 10.2.2 La boucle événementielle

[Les files d'attente de tâches](#) d' une [boucle d'événements de travail](#) ne contiennent que des événements, des rappels et une activité réseau en tant que [tâches](#) . Ces [boucles d'événements de travail](#) sont créées par l' [exécution d'un algorithme de travail](#) .

Chaque [WorkerGlobalScope](#) objet a un indicateur **de fermeture** , qui doit être initialement faux, mais qui peut être défini sur vrai par les algorithmes de la section du modèle de traitement ci-dessous.

Une fois que l' [WorkerGlobalScope](#) indicateur [de fermeture](#) de est défini sur vrai, les [files d'attente de tâches](#) de [la boucle d'événements](#) doivent supprimer toutes les [tâches](#) supplémentaires qui leur seraient ajoutées (les tâches déjà dans la file d'attente ne sont pas affectées, sauf indication contraire). En effet, une fois que l'



indicateur [de fermeture](#) est vrai, les minuteries cessent de se déclencher, les notifications pour toutes les opérations d'arrière-plan en attente sont supprimées, etc.

### 10.2.3 La durée de vie du travailleur

Les travailleurs communiquent avec d'autres travailleurs et avec [Windows](#) via [des canaux de messages](#) et leurs [MessagePort](#) objets.

Chaque *étendue globale de travailleur* [WorkerGlobalScope](#) d'objet a une liste des **ports du travailleur**, qui se compose de tous les objets qui sont intriqués avec un autre port et qui ont un (mais un seul) port appartenant à la *portée globale du travailleur*. Cette liste inclut l'implicite dans le cas des [travailleurs dédiés](#) [.MessagePortMessagePort](#).

Étant donné un [objet de paramètres d'environnement](#) *o* lors de la création ou de l'obtention d'un travailleur, le **propriétaire pertinent à ajouter** dépend du type d' [objet global](#) spécifié par *o*. Si [l'objet global](#) de *o* est un objet (c'est-à-dire si nous créons un travailleur dédié imbriqué), alors le propriétaire pertinent est cet objet global. Sinon, [l'objet global](#) de *o* est un objet et le propriétaire correspondant est celui qui est [associé à](#) [.WorkerGlobalScopeWindowWindowDocument](#).

---

Un travailleur est dit être un **travailleur autorisé** si son [WorkerGlobalScope](#) ensemble [propriétaire](#) n'est pas [vide](#) ou :

- son [jeu de propriétaires](#) a été [vide](#) pendant au plus une courte valeur de délai d'attente [définie par l'implémentation](#),
- son [WorkerGlobalScope](#) objet est un [SharedWorkerGlobalScope](#) objet (c'est-à-dire que le travailleur est un travailleur partagé), et
- l'agent utilisateur a un [navigable](#) dont [le document actif](#) n'est pas [complètement chargé](#).

*La deuxième partie de cette définition permet à un travailleur partagé de survivre pendant une courte période pendant le chargement d'une page, au cas où cette page recontacterait le travailleur partagé. Cela peut être utilisé par les agents utilisateurs comme un moyen d'éviter le coût de redémarrage d'un travailleur partagé utilisé par un site lorsque l'utilisateur navigue de page en page au sein de ce site.*

Un travailleur est dit être un **travailleur nécessaire actif** si l'un de ses [propriétaires](#) est soit [Document](#) des objets [entièrement actifs](#), soit [des travailleurs nécessaires actifs](#).

Un travailleur est dit être un **travailleur protégé** s'il s'agit d'un [travailleur nécessaire actif](#) et qu'il a des temporisateurs, des transactions de base de données ou des connexions réseau en cours, ou si sa liste des [ports du travailleur](#) n'est pas vide, ou s'il `WorkerGlobalScope` s'agit en fait d'un [SharedWorkerGlobalScope](#) objet (c'est-à-dire, le travailleur est un travailleur partagé).

Un travailleur est considéré comme un **travailleur pouvant être suspendu** s'il n'est pas un [travailleur actif nécessaire](#) , mais s'il s'agit d'un [travailleur autorisé](#) .

#### 10.2.4 Modèle de traitement

Lorsqu'un agent utilisateur doit **exécuter un worker** pour un script avec `Worker` ou `SharedWorker` objet `worker` , [URL](#) `url` , [environment settings object](#) `outside settings` , [MessagePort](#) `outside port` et un [WorkerOptions](#) dictionnaire `options` , il doit exécuter les étapes suivantes.

1. Soit *est partagé* vrai si `worker` est un [SharedWorker](#) objet, et faux sinon.
2. Laissez *le propriétaire* être le [propriétaire pertinent pour ajouter](#) des paramètres extérieurs donnés .
3. Laissez *la portée globale du travailleur parent* être nulle.
4. Si *le propriétaire* est un `WorkerGlobalScope` objet (c'est-à-dire que nous créons un travailleur dédié imbriqué), définissez *la portée globale du travailleur parent* sur *propriétaire* .
5. Soit *unsafeWorkerCreationTime* l' [heure actuelle partagée non sécurisée](#) .
6. Soit *agent* le résultat de [l'obtention d'un agent de travail dédié/partagé](#) donné *en dehors des paramètres* et *est partagé* . Exécutez le reste de ces étapes dans cet agent.
7. Supposons que *le contexte d'exécution du domaine* soit le résultat de [la création d'un nouvel agent](#) donné par le domaine et des personnalisations suivantes :
  - Pour l'objet global, si *est partagé* est vrai, créez un nouvel [SharedWorkerGlobalScope](#) objet. Sinon, créez un nouvel [DedicatedWorkerGlobalScope](#) objet.
8. Laissez *la portée globale du travailleur* être l' [objet global](#) du composant *Realm* du contexte d'exécution du domaine.

*Il s'agit de l'objet [DedicatedWorkerGlobalScope](#) OU [SharedWorkerGlobalScope](#) créé à l'étape précédente.*

9. [Configurez un objet de paramètres d'environnement de travail](#) avec un contexte d'exécution de domaine , des paramètres extérieurs et `unsafeWorkerCreationTime` , et laissez les paramètres intérieurs être le résultat.
10. Définissez [le nom](#) de la portée globale du travailleur sur la valeur du membre des options `.name`
11. [Ajouter](#) le propriétaire à [l'ensemble](#) de propriétaires de l'étendue globale du nœud de calcul .
12. Si `est partagé` est vrai, alors :
  - Définissez l' origine du [constructeur de](#) la portée globale du travailleur sur l' [origine](#) des paramètres extérieurs .
  - Définissez [l' url du constructeur](#) de la portée globale du worker sur `url` .
  - Définissez [le type](#) de la portée globale du travailleur sur la valeur du membre des options `.type`
  - Définissez [les informations d'identification](#) de la portée globale du travailleur sur la valeur du membre des options `.credentials`
13. Soit `destination` " `sharedworker`" si `est partagé` est vrai, et " `worker`" sinon.
14. Obtenez *le script* en activant la valeur du membre des options `type` :

" `classic`"

[Récupérez un script de travail classique](#) avec `url` , `outside settings` , `destination` , `inside settings` , et avec `onComplete` et `performFetch` comme défini ci-dessous.

" `module`"

[Récupérez un graphique de script de travail de module](#) en fonction de l'`url` , des paramètres extérieurs , de la destination , de la valeur du `credentials` membre options , des paramètres intérieurs et avec `onComplete` et `performFetch` comme défini ci-dessous.

Dans les deux cas, laissez `performFetch` être ce qui suit [effectuer le crochet d'extraction](#) de la requête donnée , [isTopLevel](#) et [processCustomFetchResponse](#) :

- Si `isTopLevel` a la valeur false, [récupérez](#) la requête avec [processResponseConsumeBody](#) défini sur `processCustomFetchResponse` et abandonnez ces étapes.
- Définissez [le client réservé](#) de la requête dans les paramètres internes .
- [Extraire](#) la demande avec [processResponseConsumeBody](#) défini sur les étapes suivantes en fonction de la réponse [réponse](#) et null, échec ou une [séquence d'octets](#) `bodyBytes` :

1. Définissez [l'URL](#) de la portée globale du nœud de calcul sur [l'URL](#) de la réponse .
2. [Initialisez le conteneur de stratégie de l'étendue globale de l'agent](#) en fonction de l'étendue globale de l'agent , de la réponse et des paramètres internes .
3. Si l' [initialisation Run CSP pour un](#) algorithme d' objet global renvoie " `Blocked`" lors de l' exécution sur la portée globale de l' agent , définissez la réponse sur une [erreur réseau](#) . [\[CSP\]](#)
4. Si la valeur de [la stratégie d'incorporation](#) de l' étendue globale du travailleur est [compatible avec l'isolation d'origine croisée](#) et si `Shared` est vrai, alors définissez [le mode d'isolation d'origine croisée](#) du [cluster](#) d'agents de l' agent sur " " ou " ". Celui choisi est [défini par l'implémentation](#) `.logicalconcrete`

Cela devrait vraiment être défini lors de la création du cluster d'agents, ce qui nécessite une refonte de cette section.

5. Si le résultat de [la vérification de la stratégie d'incorporation d'un objet global](#) avec la portée globale du travailleur , les paramètres extérieurs et la réponse est faux, définissez la réponse sur une [erreur réseau](#) .
6. Définissez la capacité d'isolement d' [origine croisée de l'étendue globale de l'agent](#) sur true si [le mode d'isolement d'origine croisée](#) du [cluster d'agents](#) de l' agent est " ".`concrete`
7. Si `est partagé` est faux et [que la capacité d'isolement d'origine croisée](#) du `propriétaire` est fausse, alors définissez [la capacité d'isolement d'origine croisée](#) de l'étendue globale du nœud de calcul sur faux.
8. Si `est partagé` est faux et que [le schéma](#) de [l'URL](#) de la réponse est " ", alors définissez [la capacité d'isolement d'origine croisée](#) de l'étendue globale du nœud de calcul sur faux.`data`

Il s'agit d'une valeur par défaut conservatrice pour l'instant, pendant que nous déterminons comment les travailleurs en général, et `data:` les travailleurs d'URL en particulier (qui sont d'origine croisée à partir de leur propriétaire), seront traités dans le contexte des politiques d'autorisations. Voir [w3c/webappsec-permissions-policy issue #207](#) pour plus de détails.

9. Exécutez `processCustomFetchResponse` avec `response` et `body Bytes` .

Dans les deux cas, laissez `onComplete` script donné être les étapes suivantes :

- Si *le script* est nul ou si l'erreur du *script* [à relancer](#) est non nulle, alors :
  1. [Mettre en file d'attente une tâche globale](#) sur la [source de la tâche de manipulation DOM](#) en fonction de l'[objet global pertinent](#) de *worker* pour [déclencher un événement](#) nommé *worker* [.error](#)
  2. Exécutez l' [environnement en ignorant les étapes](#) pour *les paramètres internes* .
  3. Abandonnez ces étapes.
- Associez *le travailleur* à la portée globale du travailleur .
- Laissez le port intérieur être un [nouvel](#) [MessagePort](#) objet dans [le domaine](#) des *paramètres intérieurs* .
- Associez le port intérieur à la portée globale du *nœud de calcul* .
- [Emboîtez](#) le port extérieur et le port intérieur .
- Créez un nouvel [WorkerLocation](#) objet et associez-le à la portée globale du travailleur .
- **Fermeture des travailleurs orphelins** : démarrez la surveillance du travailleur de telle sorte que dès qu'il cesse d'être un [travailleur protégé](#) et au plus tard qu'il cesse d'être un [travailleur autorisé](#) , l'indicateur [de fermeture](#) de l'étendue globale du *travailleur* est défini sur true.
- **Suspendre les travailleurs** : commencez à surveiller le travailleur, de sorte que chaque fois que l'indicateur [de fermeture](#) de la *portée globale du travailleur* est faux et que le travailleur est un [travailleur suspendable](#) , l'agent utilisateur suspend l'exécution du script dans ce travailleur jusqu'à ce que l' indicateur de [fermeture](#) passe à vrai ou le travailleur cesse d'être un [travailleur suspendable](#) .
- Définissez [l' indicateur prêt à l' exécution](#) des *paramètres à l' intérieur* .
- Si *script* est un [script classique](#) , exécutez le *script* [de script classique](#) . Sinon, c'est un [script de module](#) ; [exécutez le](#) *script* de script du module .

*En plus des possibilités habituelles de retourner une valeur ou d'échouer en raison d'une exception, cela pourrait être [prématurément interrompu](#) par l' [algorithme de terminaison d'un travailleur](#) défini ci-dessous.*

- Activer [la file d'attente des messages](#) du port extérieur .
- Si *est partagé* est faux, activez la [file d'attente de messages du port](#) du port implicite du travailleur.

- Si *est partagé* est vrai, alors [mettez en file d'attente une tâche globale](#) sur [la source de tâche de manipulation DOM](#) donnée à la portée globale du travailleur pour [déclencher un événement](#) nommé *connect* à la portée globale du travailleur, en utilisant *MessageEvent*, avec l' *data* attribut initialisé à la chaîne vide, l' *ports* attribut initialisé à un nouveau [tableau gelé](#) contenant port intérieur et l' *source* attribut initialisé à port intérieur.
- Activez la [file d'attente des messages du client](#) de l' *ServiceWorkerContainer* objet dont [le client de service worker](#) associé est [l'objet de paramètres pertinent](#) de la portée globale du worker.
- **Boucle d'événement** : exécutez la [boucle d'événement responsable](#) spécifiée par les paramètres internes jusqu'à ce qu'elle soit détruite.

*La gestion des événements ou l'exécution de rappels par [des tâches](#) exécutées par la [boucle d'événements](#) peut être [prématurément interrompue](#) par l' [algorithme de terminaison d'un travailleur](#) défini ci-dessous.*

*Le modèle de traitement de travail reste sur cette étape jusqu'à ce que la boucle d'événements soit détruite, ce qui se produit après que l' [indicateur de fermeture](#) est défini sur vrai, comme décrit dans le modèle de traitement de [boucle d'événements](#).*

- [Effacez](#) la [carte des temporisateurs actifs](#) de la portée globale de l'agent.
- Démêlez tous les ports de la liste des [ports du worker](#).
- [Ensemble de propriétaires](#) de l' *étendue globale du nœud de calcul* [vide](#).

---

Lorsqu'un agent utilisateur doit **terminer un agent**, il doit exécuter les étapes suivantes [en parallèle](#) avec la boucle principale de l'agent (le modèle de traitement « [exécuter un agent](#) » défini ci-dessus) :

1. Définissez l' [indicateur de fermeture](#) *WorkerGlobalScope* de l'objet du travailleur sur true.
2. Si des [tâches](#) sont en file d'attente dans les files d'attente de [tâches de la boucle d'événements](#) de [l'agent concerné](#) *WorkerGlobalScope* de l'objet, supprimez-les sans les traiter.
3. [Abandonnez le script](#) en cours d'exécution dans le worker.



4. WorkerGlobalScope Si l'objet du worker est en fait un DedicatedWorkerGlobalScope objet (c'est-à-dire que le worker est un worker dédié), alors videz la file d'attente des messages du port avec lequel le port implicite du worker est intriqué.

Les agents utilisateurs peuvent invoquer l'algorithme de terminaison d'un travailleur lorsqu'un travailleur cesse d'être un travailleur nécessaire actif et que le travailleur continue de s'exécuter même après que son indicateur de fermeture a été défini sur vrai.

### 10.2.5 Erreurs de script d'exécution

Chaque fois qu'une erreur de script d'exécution non détectée se produit dans l'un des scripts du travailleur, si l'erreur ne s'est pas produite lors du traitement d'une erreur de script précédente, l'agent utilisateur doit signaler l'erreur pour ce script, avec la position (numéro de ligne et numéro de colonne) où le erreur s'est produite, en utilisant l' WorkerGlobalScope objet comme cible.

Pour les nœuds de calcul partagés, si l'erreur n'est toujours pas traitée par la suite, l'erreur peut être signalée à une console de développeur.

Pour les workers dédiés, si l'erreur n'est toujours pas gérée par la suite, l'agent utilisateur doit mettre une tâche en file d'attente pour exécuter ces étapes :

1. Soit *notHandled* le résultat du déclenchement d'un événement nommé error sur l' Worker objet associé au travailleur, en utilisant ErrorEvent, avec l' cancelable attribut initialisé à true, les attributs message, filename, lineno et colno initialisés de manière appropriée, et l' error attribut initialisé à null.
2. Si *notHandled* est vrai, alors l'agent utilisateur doit agir comme si l'erreur de script d'exécution non interceptée s'était produite dans la portée globale dans laquelle se trouve l' Worker objet, répétant ainsi l'intégralité du processus de rapport d'erreur de script d'exécution d'un niveau supérieur.

Si le port implicite connectant le travailleur à son Worker objet a été désenchevêtré (c'est-à-dire si le travailleur parent a été terminé), alors l'agent utilisateur doit agir comme si l' Worker objet n'avait pas error de gestionnaire d'événements et comme si l'attribut de ce travailleur onerror était nul, mais doit autrement agir comme décrit ci-dessus.

*Ainsi, les rapports d'erreurs se propagent jusqu'à la chaîne de travailleurs dédiés jusqu'à l'original Document, même si certains des travailleurs le long de cette chaîne ont été arrêtés et les ordures récupérées.*

La [source de tâche](#) pour la tâche mentionnée ci-dessus est la [source de tâche de manipulation DOM](#) .

## 10.2.6 Création de nœuds de calcul

### 10.2.6.1 Le [AbstractWorker](#) mélange

```
interface mixin AbstractWorker {  
  
    attribute EventHandler onerror;  
  
};
```

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par les objets implémentant l' [AbstractWorker](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<a href="#">onerror</a> ✓ MDN	<a href="#">error</a>

### 10.2.6.2 Paramètres de script pour les agents

Pour **configurer un objet de paramètres d'environnement de travail** , étant donné un *contexte d'exécution* [de contexte d'exécution JavaScript](#) , un [objet de paramètres d'environnement](#) en dehors des paramètres et un nombre *unsafeWorkerCreationTime* :

1. Laissez l'origine héritée être en dehors de [l'origine](#) des paramètres .
2. Soit *realm* la valeur du composant Realm du *contexte d'exécution* .
3. Soit la portée globale du worker comme [objet global](#) du domaine .
4. Soit l'objet de paramètres un nouvel [objet de paramètres d'environnement](#) dont les algorithmes sont définis comme suit :

Le [contexte d'exécution du domaine](#)

Renvoie le contexte d'exécution .

La [carte des modules](#)



Renvoie [la carte de module](#) de l'étendue globale du travailleur .

#### L' [encodage des caractères de l'URL de l'API](#)

Renvoie [UTF-8](#) .

#### L' [URL de base de l'API](#)

Renvoie [l'URL](#) de l'étendue globale du nœud de calcul .

#### L' [origine](#)

Renvoie une [origine opaque](#) unique si [le schéma](#) de [l'URL](#) de la portée globale du travailleur est " ", et une *origine héritée* dans le cas contraire.`data`

#### Le [conteneur de politique](#)

Renvoie [le conteneur de stratégie](#) de l'étendue globale du nœud de calcul .

#### La [capacité d'isolement d'origine croisée](#)

Renvoie [la capacité d'isolement cross-origin](#) de l'étendue globale du nœud de calcul .

#### L' [origine du temps](#)

Renvoie le résultat du [grossissement de](#) `unsafeWorkerCreationTime` avec [la capacité d'isolement cross-origin](#) de la portée globale du worker .

- Définissez [l'ID](#) de l'objet de paramètres sur une nouvelle chaîne opaque unique, [l'URL de création](#) sur [l'url](#) de l'étendue globale du travailleur , [l'URL de création de niveau supérieur](#) sur null, [le contexte de navigation cible](#) sur null et [le travailleur de service actif](#) sur null.
- Si la portée globale du travailleur est un `DedicatedWorkerGlobalScope` objet, définissez [l'origine de niveau supérieur](#) de l'objet de paramètres sur [l'origine de niveau supérieur](#) des paramètres extérieurs .
- Sinon, définissez [l'origine de niveau supérieur](#) de l'objet de paramètres sur une valeur [définie par l'implémentation](#) .

Voir [Partitionnement du stockage côté client](#) pour les dernières informations sur la définition correcte de cela.

- Définissez le champ `[[HostDefined]]` du domaine sur l'objet `settings` .
- Renvoie l'objet de paramètres .

### 10.2.6.3 Travailleurs dédiés et [Worker](#) interface

```
[Exposed=(Window,DedicatedWorker,SharedWorker)]
```

```
interface Worker : EventTarget {
```

```
    constructor(USVString scriptURL, optional WorkerOptions options  
= {});
```

```
    undefined terminate();
```

```
    undefined postMessage(any message, sequence<object> transfer);
```

```
    undefined postMessage(any message, optional  
StructuredSerializeOptions options = {});
```

```
    attribute EventHandler onmessage;
```

```
    attribute EventHandler onmessageerror;
```

```
};
```

```
dictionary WorkerOptions {
```

```
    WorkerType type = "classic";
```

```
    RequestCredentials credentials = "same-origin"; // credentials
```

```
is only used if type is "module"
```

```
    DOMString name = "";
```

```
};
```

```
enum WorkerType { "classic", "module" };
```

```
Worker includes AbstractWorker;
```

```
worker = new Worker(scriptURL [, options ])
```

✓

Renvoie un nouvel [Worker](#) objet. *scriptURL* sera récupéré et exécuté en arrière-plan, créant un nouvel environnement global pour lequel *worker* représente le canal de communication. Les *options* peuvent être utilisées pour définir le [nom](#) de cet environnement global via l' *name* option, principalement à des fins de débogage. Il peut également garantir que ce nouvel environnement global prend en charge les modules JavaScript (specify *type*: "module"), et si cela est spécifié, peut également être utilisé pour spécifier comment *scriptURL* est récupéré via l' *credentials* option.

```
worker.terminate()
```

✓

Abandonne l'environnement global associé au *travailleur* .

```
worker.postMessage(message [, transfer ])
```

✓

```
worker.postMessage(message [, { transfer } ])
```

Clone le message et le transmet à l'environnement global du *travailleur* . *transfer* peut être passé sous la forme d'une liste d'objets qui doivent être transférés plutôt que clonés.

La `terminate()` méthode, lorsqu'elle est invoquée, doit entraîner l' exécution de l'algorithme de [terminaison d'un travailleur](#) sur le travailleur auquel l'objet est associé.

[Worker](#) les objets agissent comme s'ils avaient un implicite [MessagePort](#) qui leur était associé. Ce port fait partie d'un canal configuré lors de la création du nœud de calcul, mais il n'est pas exposé. Cet objet ne doit jamais être ramassé avant l' [Worker](#) objet.

Tous les messages reçus par ce port doivent être immédiatement redirigés vers l' [Worker](#) objet.

Les méthodes et sur les objets agissent comme si, lorsqu'elles étaient invoquées, elles invoquaient immédiatement le et respectif sur le port, avec les mêmes arguments, et renvoyaient la même valeur de

retour.`postMessage(message, transfer)postMessage(message, options)WorkerpostMessage(message, transfer)postMessage(message, options)`

Le `postMessage()` premier argument de la méthode peut être des données structurées :

```
worker.postMessage({opcode: 'activate', device: 1938, parameters: [23, 102]});
```

Voici les [gestionnaires d'événements](#) (et leurs [types d'événements de gestionnaire d'événements](#) correspondants ) qui doivent être pris en charge, en tant [qu'attributs IDL de gestionnaire d'événements](#) , par les objets implémentant l' [Worker](#) interface :

<a href="#">Gestionnaire d'événements</a>	<a href="#">Type d'événement du gestionnaire d'événements</a>
<code>onmessage</code>	<code>message</code>

Gestionnaire d'événements	Type d'événement du gestionnaire d'événements
<code>onmessageerror</code>	<code>messageerror</code>

Lorsque le constructeur est appelé, l'agent utilisateur doit exécuter les étapes suivantes :`Worker(scriptURL, options)`

1. L'agent utilisateur peut lancer un "`SecurityError`" `DOMException` si la demande viole une décision de politique (par exemple, si l'agent utilisateur est configuré pour ne pas autoriser la page à démarrer des travailleurs dédiés).
2. Laissez les paramètres extérieurs être l' objet de paramètres actuel .
3. Analysez l' argument `scriptURL` par rapport aux *paramètres extérieurs* .
4. Si cela échoue, lancez un "`SyntaxError`" `DOMException` .
5. Soit l'URL du nœud de calcul l' enregistrement d'URL résultant .

Toute URL de même origine (y compris `blob:` les URL) peut être utilisée. `data:` Les URL peuvent également être utilisées, mais elles créent un worker avec une origin opaque .

6. Soit `worker` un nouvel `Worker` objet.
7. Laissez le port extérieur être un nouveau domaine `MessagePort` des paramètres extérieurs .
8. Associez le port extérieur à `worker` .
9. Exécutez cette étape en parallèle :
  1. Exécutez un agent donné `worker` , URL de l'agent , paramètres extérieurs , port extérieur et options .
10. *Travailleur* de retour .

#### 10.2.6.4 Travailleurs partagés et `SharedWorker` interface



[Exposed=Window]

```
interface SharedWorker : EventTarget {
```

```
    constructor(USVString scriptURL, optional (DOMString or
```

```
    WorkerOptions) options = {});
```

```
    readonly attribute MessagePort port;
```

```
};
```

```
SharedWorker includes AbstractWorker;
```

```
sharedWorker = new SharedWorker(scriptURL [, name ])
```

✓

Renvoie un nouvel [SharedWorker](#) objet. *scriptURL* sera récupéré et exécuté en arrière-plan, créant un nouvel environnement global pour lequel *sharedWorker* représente le canal de communication. *name* peut être utilisé pour définir le [nom](#) de cet environnement global.

```
sharedWorker = new SharedWorker(scriptURL [, options ])
```

Renvoie un nouvel [SharedWorker](#) objet. *scriptURL* sera récupéré et exécuté en arrière-plan, créant un nouvel environnement global pour lequel *sharedWorker* représente le canal de communication. Les *options* peuvent être utilisées pour définir le [nom](#) de cet environnement global via l' *name* option. Il peut également garantir que ce nouvel environnement global prend en charge les modules JavaScript (specify *type*: "module"), et si cela est spécifié, peut également être utilisé pour spécifier comment *scriptURL* est récupéré via l' *credentials* option. Notez que tenter de construire un travailleur partagé avec *des options* dont les valeurs *type* ou *credentials* ne correspondent pas à un travailleur partagé existant entraînera le retour *sharedWorker* pour déclencher un événement d'erreur et ne pas se connecter au travailleur partagé existant.

```
sharedWorker.port
```

✓

Renvoie l'objet de *sharedWorker* [MessagePort](#) qui peut être utilisé pour communiquer avec l'environnement global.

L' **port** attribut doit renvoyer la valeur qui lui a été assignée par le constructeur de l'objet. Il représente le [MessagePort](#) pour communiquer avec le travailleur partagé.

Un agent utilisateur a un **gestionnaire de travail partagé** associé qui est le résultat du [démarrage d'une nouvelle file d'attente parallèle](#) .

*Chaque agent utilisateur a un seul [gestionnaire de travail partagé](#) pour plus de simplicité. Les implémentations pourraient en utiliser une par [origine](#) ; cela ne serait pas visiblement différent et permet plus de simultanéité.*

Lorsque le constructeur est appelé : `SharedWorker(scriptURL, options)`

1. En option, lancez un `"SecurityError" DOMException` si la demande viole une décision de politique (par exemple, si l'agent utilisateur est configuré pour ne pas autoriser la page à démarrer des travailleurs partagés).
2. Si `options` est un `DOMString`, définissez `options` sur un nouveau `WorkerOptions` dictionnaire dont le `name` membre est défini sur la valeur d' `options` et dont les autres membres sont définis sur leurs valeurs par défaut.
3. Laissez *les paramètres extérieurs* être l' objet de paramètres actuel .
4. Analyser `scriptURL` par rapport aux *paramètres extérieurs* .
5. Si cela échoue, lancez un `"SyntaxError" DOMException` .
6. Sinon, laissez `urlRecord` être l' enregistrement d'URL résultant .

*Toute URL de même origine (y compris blob: les URL) peut être utilisée. data: Les URL peuvent également être utilisées, mais elles créent un worker avec une origine opaque .*

7. Soit `worker` un nouvel `SharedWorker` objet.
8. Laissez *le port extérieur* être un nouveau domaine `MessagePort` des *paramètres extérieurs* .
9. Attribuez *un port extérieur* à l' `port` attribut `worker` .
10. Laissez `callerIsSecureContext` être `true` si *les paramètres extérieurs* sont un contexte sécurisé ; sinon, faux.
11. Laissez *la clé de stockage externe* être le résultat de l'exécution de l'obtention d'une clé de stockage à des fins autres que de stockage en fonction des *paramètres externes* .
12. Mettez les étapes suivantes en file d'attente dans le gestionnaire de nœuds de calcul partagé :
  1. Laissez *la portée globale du travailleur* être nulle.
  2. Pour chaque *étendue* de la liste de tous `SharedWorkerGlobalScope` les objets :
    1. Laissez *la clé de stockage de travail* être le résultat de l'exécution de l'obtention d'une clé de stockage à des fins autres que de stockage, compte tenu de l'objet de paramètres pertinent de la portée .
    2. Si toutes les conditions suivantes sont vraies :

- la clé de stockage des travailleurs est égale à la clé de stockage externe ;
- l'indicateur de fermeture de scope est faux ;
- l' url du constructeur de la portée est égale à `urlRecord` ; et
- le nom de la portée est égal à la valeur du membre de l' `optionname`

alors:

- Définissez la portée globale du travailleur sur scope .
- Pause .

*data: Les URL créent un worker avec une origine opaque . L' origine du constructeur et l'URL du constructeur sont comparées afin que la même `data:URL` puisse être utilisée dans une origine pour accéder au même `SharedWorkerGlobalScope` objet, mais ne peut pas être utilisée pour contourner la même restriction d'origine.*

3. Si la portée globale du travailleur n'est pas nulle, mais que l'agent utilisateur a été configuré pour interdire la communication entre le travailleur représenté par la portée globale du travailleur et les scripts dont l'objet de paramètres est en dehors de settings , définissez la portée globale du travailleur sur null.

*Par exemple, un agent utilisateur pourrait avoir un mode de développement qui isole un parcours de niveau supérieur particulier de toutes les autres pages, et les scripts de ce mode de développement pourraient être empêchés de se connecter aux travailleurs partagés s'exécutant en mode de navigateur normal.*

4. Si la portée globale du travailleur n'est pas nulle, vérifiez si le type et les informations d'identification de la portée globale du travailleur correspondent aux valeurs des options . Si ce n'est pas le cas, mettez une tâche en file d'attente pour déclencher un événement nommé et abandonner ces étapes.`error`
5. Si la portée globale du nœud de calcul n'est pas nulle, exécutez ces sous-sous-étapes :
  1. Laissez l'objet de paramètres être l' objet de paramètres pertinent pour la portée globale du travailleur .
  2. Laissez `workerIsSecureContext` être true si l'objet settings est un contexte sécurisé ; sinon, faux.
  3. Si `workerIsSecureContext` n'est pas `callerIsSecureContext` , placez une tâche en file d'attente pour déclencher un événement nommé `error.worker` et abandonner ces étapes. [CONTEXTES SÉCURISÉS]

4. Associez le travailleur à la portée globale du travailleur .
  5. Laissez le port intérieur être un [nouveau domaine](#)`MessagePort` de l' objet de paramètres .
  6. [Emboîtez](#) le port extérieur et le port intérieur .
  7. [Mettez une tâche en file d'attente](#) , en utilisant la [source de la tâche de manipulation DOM](#) , pour [déclencher un événement](#) nommé `connect` à la portée globale du travailleur , en utilisant `MessageEvent`, avec l' `data` attribut initialisé à la chaîne vide, l' `ports` attribut initialisé à un nouveau [tableau gelé](#) contenant uniquement le port intérieur et l' `source` attribut initialisé à à l'intérieur du port .
  8. [Ajoutez](#) le [propriétaire approprié pour ajouter](#) les paramètres externes donnés à [l'ensemble de propriétaires](#) de l'étendue globale du nœud de calcul .
6. Sinon, [en parallèle](#) , [exécutez un worker](#) donné `worker` , `urlRecord` , `outside settings` , `outside port` et `options` .

13. *Travailleur* de retour .

### 10.2.7 Capacités matérielles simultanées

```
interface mixin NavigatorConcurrentHardware {
```

```
  readonly attribute unsigned long long hardwareConcurrency;
```

```
};
```

**`self.navigator.hardwareConcurrency`**

Renvoie le nombre de processeurs logiques potentiellement disponibles pour l'agent utilisateur.

Le `navigator.hardwareConcurrency` getter de l'attribut doit renvoyer un nombre compris entre 1 et le nombre de processeurs logiques potentiellement disponibles pour l'agent utilisateur. Si cela ne peut pas être déterminé, le getter doit renvoyer 1.

Les agents utilisateurs doivent se tromper en exposant le nombre de processeurs logiques disponibles, en utilisant des valeurs inférieures uniquement dans les cas où



des limites spécifiques à l'agent utilisateur sont en place (comme une limitation du nombre de travailleurs pouvant être créés) [ou](#) lorsque l'agent utilisateur le souhaite. pour limiter les possibilités de prise d'empreintes digitales.

## 10.3 API disponibles pour les travailleurs

### 10.3.1 Importation de scripts et de bibliothèques

Les étapes de la méthode consistent à [importer des scripts dans la portée globale du travailleur](#) en fonction de [this](#) et `urls.importScripts(...urls)`

Pour **importer des scripts dans la portée globale du travailleur**, étant donné une *portée globale du travailleur* `WorkerGlobalScope` d'objet, une [liste](#) d' [urls](#) [de chaînes de valeurs scalaires](#) et une option [d'exécution du hook de récupération](#) `performFetch` :

1. Si [le type](#) de la portée globale du travailleur est " ", lève une exception `moduleTypeError`
2. Soit l'objet de paramètres l' [objet de paramètres actuel](#) .
3. Si `urls` est vide, retour.
4. [Analysez](#) chaque valeur dans les URL par rapport à l'objet de paramètres . En cas d'échec, jetez un `"SyntaxError"` `DOMException` .
5. Pour chaque URL dans les [enregistrements d'URL résultants](#) :
  1. [Récupérez un script classique importé par le travailleur](#) en fonction de l'URL et de l'objet de paramètres , en transmettant `performFetch` s'il est fourni. Si cela réussit, laissez `script` être le résultat. Sinon, relancez l'exception.
  2. *Exécutez le script* [script classique](#) , avec l'argument `rethrow errors` défini sur `true`.

*Le script s'exécutera jusqu'à ce qu'il revienne, échoue à analyser, échoue à intercepter une exception ou soit [prématurément abandonné](#) par l'algorithme [de terminaison d'un travailleur](#) défini ci-dessus.*

Si une exception a été levée ou si le script a été [abandonné prématurément](#) , abandonnez toutes ces étapes, laissant l'exception ou l'abandon continuer à être traité par le [script](#) appelant .

*Service Workers est un exemple de spécification qui exécute cet algorithme avec son propre [hook fetch](#) . [SW]*

### 10.3.2 L' WorkerNavigator interface



L' **navigator** attribut de l' WorkerGlobalScope interface doit renvoyer une instance de l' WorkerNavigator interface, qui représente l'identité et l'état de l'agent utilisateur (le client) :

```
[Exposed=Worker]
```

```
interface WorkerNavigator {};
```

```
WorkerNavigator includes NavigatorID;
```

```
WorkerNavigator includes NavigatorLanguage;
```

```
WorkerNavigator includes NavigatorOnLine;
```

```
WorkerNavigator includes NavigatorConcurrentHardware;
```

### 10.3.3 L' WorkerLocation interface



```
[Exposed=Worker]
```

```
interface WorkerLocation {
```

```
stringifier readonly attribute USVString href;
```

```
readonly attribute USVString origin;
```

```
readonly attribute USVString protocol;
```

```
readonly attribute USVString host;
```

```
readonly attribute USVString hostname;
```

```
readonly attribute USVString port;
```

```
readonly attribute USVString pathname;
```

```
readonly attribute USVString search;
```

```
readonly attribute USVString hash;
```

```
};
```

Un `WorkerLocation` objet a un `WorkerGlobalScope` objet associé (un `WorkerGlobalScope` objet).

✓ MDN

Les `href` étapes du getter consistent à renvoyer l' `url` de `cet WorkerGlobalScope objet` , `sérialisé` .

✓ MDN

Les étapes du getter `consistentorigin` à retourner la `sérialisation` de `l'origine` de l' url `de cet` objet `.WorkerGlobalScope`

✓ MDN

Les `protocol` étapes du getter consistent à renvoyer `le schéma` de l' `URL` de `cet` objet , suivi de " `WorkerGlobalScope` " .:

✓ MDN

Les `host` étapes du getter sont :

1. Soit `url` l' `url` de `cet WorkerGlobalScope objet` . \_
2. Si `l'hôte` de l' `url` est nul, renvoie la chaîne vide.
3. Si `le port` de l' `url` est nul, renvoie `l'hôte` de l' `url` , `sérialisé` .
4. Renvoie `l'hôte` de l' `url` , `sérialisé` , suivi de " " et `le port` de l' `url` , `sérialisé` .:

✓ MDN

Les `hostname` étapes du getter sont :

1. Soit `host` l' `hôte` de l' `url` de `cet WorkerGlobalScope objet` .
2. Si `host` est null, renvoie la chaîne vide.
3. `Hôte` de retour , `sérialisé` .

✓ MDN

Les **port**étapes du getter sont :

1. Soit *port* le [port](#) de l' [url](#) de [cet WorkerGlobalScope objet](#) .
2. Si *port* est nul, renvoie la chaîne vide.
3. *Port* de retour , [sérialisé](#) .



Les **pathname**étapes du getter consistent à renvoyer le résultat du [chemin d'URL sérialisant](#) l' [url](#) de [cet WorkerGlobalScope objet](#) .



Les **search**étapes du getter sont :

1. Soit *query* [la](#) requête de l' [url](#) de [cet WorkerGlobalScope objet](#) .
2. Si *la requête* est nulle ou la chaîne vide, renvoie la chaîne vide.
3. Renvoie " ?", suivi de *query* .



Les **hash**étapes du getter sont :

1. Soit *fragment* le [fragment](#) de l' [url](#) de [cet WorkerGlobalScope objet](#) .
2. Si *fragment* est null ou la chaîne vide, renvoie la chaîne vide.
3. Renvoie " #", suivi de *fragment* .

## 1. [11 Worklets](#)

### 1. [11.1 Présentation](#)

1. [11.1.1 Motivations](#)
2. [11.1.2 Code idempotence](#)
3. [11.1.3 Évaluation spéculative](#)

### 2. [11.2 Exemples](#)

1. [11.2.1 Chargement des scripts](#)
2. [11.2.2 Enregistrement d'une classe et invocation de ses méthodes](#)

### 3. [11.3 Infrastructures](#)

1. [11.3.1 Le périmètre mondial](#)

1. [11.3.1.1 Agents et boucles d'événements](#)
2. [11.3.1.2 Création et résiliation](#)
3. [11.3.1.3 Paramètres de script pour les worklets](#)
2. [11.3.2 La `Worklet` classe](#)
3. [11.3.3 Durée de vie du worklet](#)

## 11 Worklets

### 11.1 Présentation

*Cette section est non normative.*

Les worklets sont un élément d'infrastructure de spécification qui peut être utilisé pour exécuter des scripts indépendamment de l'environnement d'exécution JavaScript principal, sans nécessiter de modèle d'implémentation particulier.

L'infrastructure de worklet spécifiée ici ne peut pas être utilisée directement par les développeurs Web. Au lieu de cela, d'autres spécifications s'en inspirent pour créer des types de worklets directement utilisables, spécialisés pour exécuter des parties particulières du pipeline d'implémentation du navigateur.

#### 11.1.1 Motivations

*Cette section est non normative.*

Il est difficile d'autoriser des points d'extension au rendu ou à d'autres parties sensibles du pipeline de mise en œuvre, telles que la sortie audio. Si les points d'extension étaient réalisés avec un accès complet aux API disponibles sur `Window`, les moteurs devraient abandonner les hypothèses précédemment retenues sur ce qui pourrait se passer au milieu de ces phases. Par exemple, lors de la phase de mise en page, les moteurs de rendu supposent qu'aucun DOM ne sera modifié.

De plus, la définition de points d'extension dans l' `Window` environnement limiterait les agents utilisateurs à effectuer un travail dans le même thread que l' `Window` objet. (À moins que les implémentations aient ajouté une infrastructure complexe et à forte surcharge pour permettre des API thread-safe, ainsi que des garanties de jonction de threads.)

Les worklets sont conçus pour permettre des points d'extension, tout en conservant les garanties sur lesquelles les agents utilisateurs s'appuient actuellement. Cela se

fait via de nouveaux environnements globaux, basés sur des sous-classes de [WorkletGlobalScope](#).

Les worklets sont similaires aux web workers. Cependant, ils :

- Sont indépendants des threads. C'est-à-dire qu'ils ne sont pas conçus pour s'exécuter sur un thread séparé dédié, comme chaque travailleur. Les implémentations peuvent exécuter des worklets où elles le souhaitent (y compris sur le thread principal).
- Sont capables d'avoir plusieurs instances en double de la portée globale créées, à des fins de parallélisme.
- N'utilisez pas d'API basée sur les événements. Au lieu de cela, les classes sont enregistrées sur la portée globale, dont les méthodes sont invoquées par l'agent utilisateur.
- Avoir une surface d'API réduite sur le périmètre global.
- Avoir une durée de vie pour leur [objet global](#) qui est définie par d'autres spécifications, souvent d'une manière [définie par l'implémentation](#) .

Comme les worklets ont des frais généraux relativement élevés, il est préférable de les utiliser avec parcimonie. Pour cette raison, une donnée [WorkletGlobalScope](#) devrait être partagée entre plusieurs scripts distincts. (Ceci est similaire à la façon dont un seul [Window](#) est partagé entre plusieurs scripts distincts.)

Les worklets sont une technologie générale qui sert différents cas d'utilisation. Certains worklets, tels que ceux définis dans *CSS Painting API* , fournissent des points d'extension destinés aux calculs sans état, idempotents et de courte durée, qui ont des considérations particulières, comme décrit dans les deux sections suivantes. D'autres, tels que ceux définis dans *Web Audio API* , sont utilisés pour les opérations de longue durée avec état. [\[CSSPAINT\]](#) [\[WEBAUDIO\]](#)

### 11.1.2 Code idempotence

Certaines spécifications qui utilisent des worklets sont destinées à permettre aux agents utilisateurs de paralléliser le travail sur plusieurs threads ou de déplacer le travail entre les threads selon les besoins. Dans ces spécifications, les agents utilisateurs peuvent invoquer des méthodes sur une classe fournie par un développeur Web dans un ordre [défini par l'implémentation](#) .

Par conséquent, pour éviter les problèmes d'interopérabilité, les auteurs qui enregistrent des classes sur de tels [WorkletGlobalScope](#) doivent rendre leur code idempotent. Autrement dit, une méthode ou un ensemble de méthodes sur la classe doit produire la même sortie compte tenu d'une entrée particulière.

Cette spécification utilise les techniques suivantes afin d'encourager les auteurs à écrire du code de manière idempotente :

- Aucune référence à l'objet global n'est disponible (c'est-à-dire qu'il n'y a pas d'équivalent à `self` ou `WorkletGlobalScope`).

Bien que cela ait été l'intention lorsque les worklets ont été spécifiés pour la première fois, l'introduction de `globalThis` l'a rendu caduque. Voir [le numéro 6059](#) pour plus de discussion.

- Le code est chargé en tant que [script de module](#) , ce qui entraîne l'exécution du code en mode strict et sans `this` référence partagée au proxy global.

Ensemble, ces restrictions permettent d'empêcher deux scripts différents de partager l'état à l'aide des propriétés de l' [objet global](#) .

De plus, les spécifications qui utilisent des worklets et ont l'intention d'autoriser un comportement [défini par l'implémentation](#) doivent respecter les règles suivantes :

- Ils doivent exiger que les agents utilisateurs aient toujours au moins deux `WorkletGlobalScope` instances par `Worklet`, et assignent au hasard une méthode ou un ensemble de méthodes sur une classe à une `WorkletGlobalScope` instance particulière. Ces spécifications peuvent fournir une option de non-participation sous des contraintes de mémoire.
- Ces spécifications doivent permettre aux agents utilisateurs de créer et de détruire `WorkletGlobalScope` à tout moment des instances de leurs sous-classes.

### 11.1.3 Évaluation spéculative

Certaines spécifications qui utilisent des worklets peuvent invoquer des méthodes sur une classe fournie par un développeur Web en fonction de l'état de l'agent utilisateur. Pour augmenter la concurrence entre les threads, un agent utilisateur peut invoquer une méthode de manière spéculative, en fonction d'états futurs potentiels.

Dans ces spécifications, les agents utilisateurs peuvent invoquer de telles méthodes à tout moment, et avec n'importe quels arguments, pas seulement ceux correspondant à l'état actuel de l'agent utilisateur. Les résultats de ces évaluations spéculatives ne sont pas affichés immédiatement, mais peuvent être mis en cache pour être utilisés si l'état de l'agent utilisateur correspond à l'état spéculé. Cela peut augmenter la simultanéité entre l'agent utilisateur et les threads de worklet.

Par conséquent, pour éviter les risques d'interopérabilité entre les agents utilisateurs, les auteurs qui enregistrent des classes sur de tels `WorkletGlobalScope`s devraient

rendre leur code sans état. Autrement dit, le seul effet de l'appel d'une méthode doit être son résultat, et non des effets secondaires tels que la mise à jour de l'état mutable.

Les mêmes techniques qui encouragent [l'idempotence du code](#) encouragent également les auteurs à écrire du code sans état.

## 11.2 Exemples

*Cette section est non normative.*

Pour ces exemples, nous utiliserons un faux worklet. L' [Window](#) objet fournit deux [Worklet](#) instances, qui exécutent chacune du code dans leur propre collection de [FakeWorkletGlobalScope](#) :

```
partial interface Window {  
  
    [SameObject, SecureContext] readonly attribute Worklet  
    fakeWorklet1;  
  
    [SameObject, SecureContext] readonly attribute Worklet  
    fakeWorklet2;  
  
};
```

Chacun [Window](#) a deux [Worklet](#) instances, **faux worklet 1** et **faux worklet 2**. Les deux ont leur [type de portée globale de worklet](#) défini sur [FakeWorkletGlobalScope](#), et leur [type de destination de worklet](#) défini sur "fakeworklet". Les agents utilisateurs doivent créer au moins deux [FakeWorkletGlobalScope](#) instances par worklet.

"fakeworklet" n'est pas réellement une [destination](#) valide par Fetch. Mais cela illustre comment les vrais worklets auraient généralement leur propre destination spécifique au type de worklet. [\[ALLER CHERCHER\]](#)

Les [fakeWorklet1](#) étapes du getter consistent à renvoyer [ce](#) faux worklet [1](#).

Les [fakeWorklet2](#) étapes du getter consistent à renvoyer [ce](#) faux worklet [2](#).

---



```

[Global=(Worklet,FakeWorklet),
Exposed=FakeWorklet,
SecureContext]

interface FakeWorkletGlobalScope : WorkletGlobalScope {

  undefined registerFake(DOMString type, Function
classConstructor);

};

```

Chacun `FakeWorkletGlobalScope` a une **carte de constructeurs de classe enregistrée**, qui est une [carte ordonnée](#), initialement vide.

Les étapes de la méthode consistent à définir [les](#) constructeurs de [classe enregistrés](#) [map](#) [ *type* ] sur `classConstructor.registerFake(type, classConstructor)`

### 11.2.1 Chargement des scripts

*Cette section est non normative.*

Pour charger des scripts dans [le faux worklet 1](#), un développeur Web écrirait :

```

window.fakeWorklet1.addModule('script1.mjs');
window.fakeWorklet1.addModule('script2.mjs');

```

Notez que le script qui termine la récupération et s'exécute en premier dépend de la synchronisation du réseau : il peut s'agir de `script1.mjs` ou `script2.mjs`. Cela n'a généralement pas d'importance pour les scripts bien écrits destinés à être chargés dans des worklets, s'ils suivent les suggestions sur la préparation de [l'évaluation spéculative](#).

Si un développeur Web souhaite effectuer une tâche uniquement après que les scripts ont été exécutés et chargés avec succès dans certains worklets, il peut écrire :

```

Promise.all([
  window.fakeWorklet1.addModule('script1.mjs'),
  window.fakeWorklet2.addModule('script2.mjs')
]).then(() => {
  // Do something which relies on those scripts being loaded.

```

```
});
```

---

Un autre point important concernant le chargement de scripts est que les scripts chargés peuvent être exécutés dans plusieurs [WorkletGlobalScope](#)s per [Worklet](#), comme indiqué dans la section sur [l'idempotence du code](#) . En particulier, la spécification ci-dessus pour [le faux worklet 1](#) et [le faux worklet 2](#) l'exige. Alors, considérez un scénario tel que le suivant :

```
// script.mjs
console.log("Hello from a FakeWorkletGlobalScope!");
// app.mjs
window.fakeWorklet1.addModule("script.mjs");
```

Cela pourrait entraîner une sortie telle que la suivante à partir de la console d'un agent utilisateur :

```
[fakeWorklet1#1] Hello from a FakeWorkletGlobalScope!
[fakeWorklet1#4] Hello from a FakeWorkletGlobalScope!
[fakeWorklet1#2] Hello from a FakeWorkletGlobalScope!
[fakeWorklet1#3] Hello from a FakeWorkletGlobalScope!
```

Si l'agent utilisateur décidait à un moment donné de tuer et de redémarrer la troisième instance de [FakeWorkletGlobalScope](#), la console imprimerait à nouveau `[fakeWorklet1#3] Hello from a FakeWorkletGlobalScope!` lorsque cela se produirait.

### 11.2.2 Enregistrement d'une classe et invocation de ses méthodes

*Cette section est non normative.*

Disons que l'une des utilisations prévues de notre faux worklet par les développeurs Web est de leur permettre de personnaliser le processus très complexe de négation booléenne. Ils peuvent enregistrer leur personnalisation comme suit :

```
// script.mjs
registerFake('negation-processor', class {
  process(arg) {
    return !arg;
  }
});
```

```
});
// app.mjs
window.fakeWorklet1.addModule("script.mjs");
```

Pour utiliser de telles classes enregistrées, la spécification des faux worklets pourrait définir un algorithme **de recherche de l'opposé du vrai**, étant donné un `Worklet` `worklet` :

1. Créez éventuellement une étendue globale de worklet pour `worklet`.
2. Soit `workletGlobalScope` l'une des étendues globales de `worklet`, choisie d'une manière définie par l'implémentation.
3. Soit `classConstructor` la map [" "] des constructeurs de classe enregistrés de `workletGlobalScope`.`negation-processor`
4. Soit `classInstance` le résultat de la construction de `classConstructor`, sans arguments.
5. Soit *fonction* `Get ( classInstance , " ")`. Renvoyez toutes les exceptions.`process`
6. Supposons que *le rappel* soit le résultat de la conversion de la fonction en une `Function`instance Web IDL.
7. Renvoie le résultat de l'invocation de *callback* avec les arguments « true » et avec `classInstance` comme callback this value.

*Une autre architecture de spécification, peut-être meilleure, consisterait à extraire la `process`propriété " " et à la convertir en a `Function`au moment de l'enregistrement, dans le cadre des `registerFake()` étapes de la méthode.*

## 11.3 Infrastructures

### 11.3.1 Le périmètre mondial

Les sous-classes de `WorkletGlobalScope` sont utilisées pour créer des objets globaux dans lesquels le code chargé dans un particulier `Worklet` peut s'exécuter.

```
[Exposed=Worklet, SecureContext]
```

```
interface WorkletGlobalScope {};
```

*D'autres spécifications sont destinées à sous-classer `WorkletGlobalScope`, en ajoutant des API pour enregistrer une classe, ainsi que d'autres API spécifiques à leur type de worklet.*

Chacun [WorkletGlobalScope](#) a une **carte de module** associée . Il s'agit d'une [carte de module](#) , initialement vide.

### 11.3.1.1 Agents et boucles d'événements

*Cette section est non normative.*

Chacun [WorkletGlobalScope](#) est contenu dans son propre [agent de worklet](#) , qui a [sa boucle d'événement](#) correspondante . Cependant, dans la pratique, la mise en œuvre de ces agents et boucles d'événements devrait être différente de la plupart des autres.

Un [agent de worklet](#) existe pour chacun [WorkletGlobalScope](#) puisque, en théorie, une implémentation pourrait utiliser un thread séparé pour chaque [WorkletGlobalScope](#) instance, et permettre ce niveau de parallélisme est mieux fait en utilisant des agents. Cependant, comme leur valeur `[[CanBlock]]` est fausse, il n'est pas nécessaire que les agents et les threads soient un à un. Cela donne aux implémentations la liberté d'exécuter des scripts chargés dans un worklet sur n'importe quel thread, y compris un code en cours d'exécution d'autres agents avec `[[CanBlock]]` de false, comme le thread d'un agent de fenêtre d'origine similaire ("le thread principal " ). Comparez cela avec [les agents de travail dédiés](#) , dont la vraie valeur pour `[[CanBlock]]` nécessite effectivement qu'ils obtiennent un thread de système d'exploitation dédié.

[Les boucles d'événements](#) de worklet sont également quelque peu spéciales. Ils ne sont utilisés que pour [les tâches](#) associées à [addModule\(\)](#) , les tâches dans lesquelles l'agent utilisateur invoque des méthodes définies par l'auteur et [les microtâches](#) . Ainsi, même si le [modèle de traitement des boucles d'événements](#) spécifie que toutes les boucles d'événements s'exécutent en continu, les implémentations peuvent obtenir des résultats observables équivalents à l'aide d'une stratégie plus simple, qui [appelle](#) simplement les méthodes fournies par l'auteur, puis s'appuie sur ce processus pour [effectuer un point de contrôle de microtâche](#) .

### 11.3.1.2 Création et résiliation

Pour **créer une étendue globale de worklet** pour un [Worklet](#) *worklet* :

1. Laissez *outsideSettings* être [l'objet de paramètres pertinent](#) du *worklet* .
2. Soit *agent* le résultat de [l'obtention d'un agent de worklet](#) donné *outsideSettings* . Exécutez le reste de ces étapes dans cet agent.

3. Soit *realmExecutionContext* le résultat de [la création d'un nouvel agent](#) donné par le domaine et des personnalisations suivantes :
  - Pour l'objet global, créez un nouvel objet du type donné par worklet *global* [scope type](#) .
4. Soit *workletGlobalScope* l' [objet global](#) du composant Realm de *realmExecutionContext* .
5. Soit *insideSettings* le résultat de [la configuration d'un objet de paramètres d'environnement de worklet](#) donné *realmExecutionContext* et *outsideSettings* .
6. Laissez *pendingAddedModules* être un [clone](#) de [la liste des modules ajoutés](#) du *worklet* .
7. Soit *runNextAddedModule* les étapes suivantes :
  - Si *pendingAddedModules* [n'est pas vide](#) , alors :
    1. Laissez *moduleURL* être le résultat de [la sortie de la file d'attente](#) de *pendingAddedModules* .
    2. [Récupérer un graphique de script de worklet](#) donné *moduleURL* , *insideSettings* , [type de destination de worklet](#) de *worklet* , [quel mode d'informations d'identification?](#) , *insideSettings* , la carte des réponses du [module](#) *worklet* , et avec les étapes suivantes du *script* donné :
 

*Cela n'effectuera pas réellement de requête réseau, car il réutilisera simplement [les réponses de la mappe de réponses](#) du module de worklet . L'objectif principal de cette étape est de créer un nouveau [script de module](#) spécifique à *workletGlobalScope* à partir de la [réponse](#) .*

      1. [Assert](#) : le *script* n'est pas nul, car la récupération a réussi et le texte source a été analysé avec succès lorsque [la carte des réponses](#) du module de *worklet* a été initialement remplie avec *moduleURL* .
      2. [Exécutez un script de module](#) *script* donné .
      3. Exécutez *runNextAddedModule* .
  - 3. Abandonnez ces étapes.
  - [Ajoutez](#) *workletGlobalScope* aux [étendues globales de worklet](#) associées à l' objet [global](#) *outsideSettings* .[Document](#)
  - [Ajoutez](#) *workletGlobalScope* aux [portées globales](#) du *worklet* .

- Exécutez la [boucle d'événement responsable](#) spécifiée par *insideSettings* .

8. Exécutez *runNextAddedModule* .

Pour **mettre fin à une portée globale de worklet** avec un [WorkletGlobalScope](#) *workletGlobalScope* :

1. Soit *eventLoop* la [boucle d'événement pertinente](#) de l'agent *workletGlobalScope* .
2. S'il y a des [tâches](#) en file d'attente dans [les files d'attente de tâches](#) d' *eventLoop* , supprimez-les sans les traiter.
3. Attendez que *eventLoop* termine la [tâche en cours d'exécution](#) .
4. Si l'étape précédente ne se termine pas dans un délai [défini par l'implémentation , abandonnez le script](#) en cours d'exécution dans le worklet.
5. Détruisez *eventLoop* .
6. [Supprimez](#) *workletGlobalScope* des [étendues globales](#) de [Worklet](#) dont [les étendues globales](#) contiennent *workletGlobalScope* .
7. [Supprimez](#) *workletGlobalScope* des [étendues globales de workletDocument](#) dont les [étendues globales de worklet](#) contiennent *workletGlobalScope* .

### 11.3.1.3 Paramètres de script pour les worklets

Pour **configurer un objet de paramètres d'environnement de worklet** , étant donné un [contexte d'exécution JavaScript](#) *executionContext* et un [objet de paramètres d'environnement](#) *outsideSettings* :

1. Soit *origin* une unique [origine opaque](#) .
2. Laissez *legacyAPIBaseURL* être [l'URL de base de l'API](#) de *outsideSettings* .
3. Laissons *legacyPolicyContainer* être un [clone](#) du [conteneur](#) de stratégie de *outsideSettings* .
4. Soit *realm* la valeur du composant Realm de *executionContext* .
5. Soit *workletGlobalScope* l' [objet global](#) du domaine .
6. Soit *settingsObject* un nouvel [objet de paramètres d'environnement](#) dont les algorithmes sont définis comme suit :

Le [contexte d'exécution du domaine](#)

Retourne `executionContext` .

La [carte des modules](#)

Renvoie [la carte des modules](#) de `workletGlobalScope` .

L' [encodage des caractères de l'URL de l'API](#)

Renvoie [UTF-8](#) .

L' [URL de base de l'API](#)

Renvoie `l'APIBaseURL` héritée .

*Contrairement aux workers ou à d'autres variables globales dérivées d'une seule ressource, les worklets n'ont pas de ressource principale ; à la place, plusieurs scripts, chacun avec sa propre URL, sont chargés dans la portée globale via `worklet.addModule()` . Donc, cette [URL de base de l'API](#) est plutôt différente de celle des autres globales. Cependant, jusqu'à présent, cela n'a pas d'importance, car aucune API disponible pour le code de worklet n'utilise l' [URL de base de l'API](#) .*

L' [origine](#)

`Origine` du retour .

Le [conteneur de politique](#)

Retourne `legacyPolicyContainer` .

La [capacité d'isolement d'origine croisée](#)

Retour **À FAIRE** .

L' [origine du temps](#)

[Assert](#) : cet algorithme n'est jamais appelé, car l' [origine temporelle](#) n'est pas disponible dans un contexte de worklet.

7. Définissez l' [id](#) de `settingsObject` sur une nouvelle chaîne opaque unique, l' [URL de création](#) sur `legacyAPIBaseURL` , l' [URL de création de niveau supérieur](#) sur null, l' [origine de niveau supérieur](#) sur l' [origine de niveau supérieur](#) de `outsideSettings` , le [contexte de navigation cible](#) sur null et le [travailleur de service actif](#) sur null .
8. Définissez le `champ` `[[HostDefined]]` du domaine sur `settingsObject` .
9. Renvoie `settingsObject` .

### 11.3.2 La [Worklet](#) classe

La [Worklet](#) classe offre la possibilité d'ajouter des scripts de module dans ses [WorkletGlobalScope](#)s associés. L'agent utilisateur peut alors créer des classes enregistrées sur le [WorkletGlobalScope](#) et invoquer leurs méthodes.

```
[Exposed=Window, SecureContext]

interface Worklet {

    [NewObject] Promise<undefined> addModule(USVString moduleURL,
    optional WorkletOptions options = {});

};

dictionary WorkletOptions {

    RequestCredentials credentials = "same-origin";

};
```

Les spécifications qui créent [Worklet](#) des instances doivent spécifier les éléments suivants pour une instance donnée :

- son **type de portée globale worklet** , qui doit être un type Web IDL qui [hérite](#) de [WorkletGlobalScope](#); et
- son **type de destination de worklet** , qui doit être une [destination](#) , et est utilisé lors de la récupération des scripts.

```
await worklet.addModule(moduleURL[, { credentials }])
```

✓

Charge et exécute le [script de module](#) donné par *moduleURL* dans toutes les [portées globales](#) du *worklet* . Il peut également créer des étendues globales supplémentaires dans le cadre de ce processus, selon le type de worklet. La promesse renvoyée sera remplie une fois que le script aura été chargé et exécuté avec succès dans toutes les étendues globales.

L' [credentials](#) option peut être définie sur un [mode d'identification](#) pour modifier le processus de récupération de script. Sa valeur par défaut est " same-origin".

Tout échec de [récupération](#) du script ou de ses dépendances entraînera le rejet de la promesse renvoyée avec un " [AbortError](#) " [DOMException](#) . Toute erreur d'analyse du script ou de ses dépendances entraînera le rejet de la promesse renvoyée avec l'exception générée lors de l'analyse.



A `Worklet` a une [liste](#) de **portées globales** , qui contient des instances du [type de portée globale](#) `worklet` de `worklet` de . Il est initialement vide.

A `Worklet` a une **liste de modules ajoutés** , qui est une [liste](#) d' [URL](#) , initialement vide. L'accès à cette liste doit être thread-safe.

A `Worklet` a une **carte de réponses de module** , qui est une [carte ordonnée](#) des [URL](#) vers " `fetching`" ou [des tuples](#) consistant en une [réponse](#) et soit une valeur nulle, un échec ou une [séquence d'octets](#) représentant le corps de la réponse. Cette carte est initialement vide et son accès doit être thread-safe.

*La [liste des modules ajoutés](#) et la [carte des réponses des modules](#) existent pour garantir que `WorkletGlobalScope` les s créés à des moments différents obtiennent [des scripts de module](#) équivalents exécutés en eux, basés sur le même texte source. Cela permet à la création de `WorkletGlobalScope`s supplémentaires d'être transparente pour l'auteur.*

*En pratique, les agents utilisateurs ne sont pas censés implémenter ces structures de données, et les algorithmes qui les consultent, en utilisant des techniques de programmation thread-safe. Au lieu de cela, lorsque `addModule()` est appelé, les agents utilisateurs peuvent récupérer le graphe du module sur le thread principal et envoyer le texte source récupéré (c'est-à-dire les données importantes contenues dans la [carte des réponses du module](#) ) à chaque thread qui a un `WorkletGlobalScope`.*

*Ensuite, lorsqu'un agent utilisateur [crée](#) un `new WorkletGlobalScope` pour un donné `Worklet`, il peut simplement envoyer la carte du texte source récupéré et la liste des points d'entrée du thread principal au thread contenant le `new WorkletGlobalScope`.*

Les étapes de la méthode sont : `addModule(moduleURL, options)`

1. Laissez `outsideSettings` être l' [objet de paramètres pertinent](#) de [this](#) .
2. [Analyser](#) `moduleURL` par rapport à `outsideSettings` .
3. Si cela échoue, renvoyez [une promesse rejetée](#) avec un `"SyntaxError"` `DOMException` .
4. Soit `moduleURLRecord` l' [enregistrement d'URL résultant](#) .
5. Que `la promesse` soit une nouvelle promesse.
6. Exécutez les étapes suivantes [en parallèle](#) :
  1. Si [ce](#) champ d' [application global est vide](#) , alors :
    1. [Créez une étendue globale de worklet](#) en fonction de [ceci](#) .

2. Facultativement, [créez](#) des instances de portée globale supplémentaires étant donné [this](#) , en fonction du worklet spécifique en question et de ses spécifications.
3. Attendez que toutes les étapes du ou des processus [de création](#) , y compris celles qui se déroulent au sein des [agents de worklet](#) , soient terminées avant de poursuivre.
2. Soit `pendingTasks` la [taille](#) de [cette](#) étendue [globale](#) .
3. Soit `addedSuccessfully` faux.
4. [Pour chaque](#) `workletGlobalScope` des [portées globales](#) de [this](#) , [mettez en file d'attente une tâche globale](#) sur la [source de tâche réseau](#) donnée à `workletGlobalScope` pour [récupérer un graphique de script de worklet](#) donné `moduleURLRecord` , `outsideSettings` , [this](#) 's [worklet type de destination](#) , `options [ " " ]` , [les paramètres pertinents](#) de `workletGlobalScope` [objet](#) , [c'est](#) la carte des réponses du [module](#) , et les étapes suivantes étant donné *le script* `credentials`:

*Seule la première de ces extractions effectuera réellement une requête réseau ; ceux des autres `WorkletGlobalScope` réutiliseront [les réponses de la carte des réponses de ce module](#) .*

1. Si *le script* est nul, alors :
  1. [Mettez en file d'attente une tâche globale](#) sur la [source de tâche réseau](#) en fonction [de](#) l' [objet global pertinent](#) pour effectuer les étapes suivantes :
    1. Si `pendingTasks` n'est pas -1, alors :
      1. Définissez *les tâches en attente* sur -1.
      2. Rejeter *la promesse* avec un `"AbortError" DOMException` .
    2. Abandonnez ces étapes.
  2. Si [l'erreur de relance](#) du *script* n'est pas nulle, alors :
    1. [Mettez en file d'attente une tâche globale](#) sur la [source de tâche réseau](#) en fonction [de](#) l' [objet global pertinent](#) pour effectuer les étapes suivantes :
      1. Si `pendingTasks` n'est pas -1, alors :
        1. Définissez *les tâches en attente* sur -1.
        2. Rejeter *la promesse* avec l'erreur du *script à relancer* .

2. Abandonnez ces étapes.
3. Si *addedSuccessfully* est faux, alors :
  1. [Ajouter](#) *moduleURLRecord* à [cette](#) liste de [modules ajoutés](#) .
  2. Définissez *addedSuccessfully* sur true.
4. [Exécutez un script de module](#) *script* donné .
5. [Mettez en file d'attente une tâche globale](#) sur la [source de tâche réseau](#) en fonction [de](#) l' [objet global pertinent](#) pour effectuer les étapes suivantes :
  1. Si *pendingTasks* n'est pas -1, alors :
    1. Définissez *pendingTasks* sur *pendingTasks* - 1.
    2. Si *pendingTasks* vaut 0, alors résolvez *promise* .
7. *Promesse* de retour .

### 11.3.3 Durée de vie du worklet

La durée de vie de a [Worklet](#) n'a pas de considérations particulières ; il est lié à l'objet auquel il appartient, tel que le [Window](#).

Chacun [Document](#) a un **worklet global scopes** , qui est un [ensemble](#) de [WorkletGlobalScopes](#), initialement vide.

La durée de vie de a [WorkletGlobalScope](#) est, au minimum, liée aux [portées globales du worklet](#) [Document](#) dont elle contient. En particulier, [la destruction](#) du mettra [fin](#) au correspondant et permettra son ramasse-miettes. [DocumentWorkletGlobalScope](#)

De plus, les agents utilisateurs peuvent, à tout moment, [mettre fin](#) à un [WorkletGlobalScope](#), sauf si la spécification définissant le type de worklet correspondant indique le contraire. Par exemple, ils peuvent les terminer si la [boucle d'événements](#) de [l'agent de worklet](#) n'a pas [de tâches](#) en file d'attente, ou si l'agent utilisateur n'a aucune opération en attente prévoyant d'utiliser le worklet, ou si l'agent utilisateur détecte des opérations anormales telles que des boucles infinies ou rappels dépassant les délais imposés.

Enfin, les spécifications pour des types de worklets spécifiques peuvent donner des détails plus spécifiques sur le moment de [créer](#) [WorkletGlobalScope](#) des s pour un type de worklet donné. Par exemple, ils peuvent les créer lors de processus spécifiques qui font appel au code de worklet, comme dans l' [exemple](#) .

## 1. [12 Stockage Web](#)

1. [12.1 Présentation](#)
2. [12.2 L'API](#)
  1. [12.2.1 L'`Storage` interface](#)
  2. [12.2.2 Le `sessionStorage` getter](#)
  3. [12.2.3 Le `localStorage` getter](#)
  4. [12.2.4 L'`StorageEvent` interface](#)
3. [12.3 Confidentialité](#)
  1. [12.3.1 Suivi des utilisateurs](#)
  2. [12.3.2 Sensibilité des données](#)
4. [12.4 Sécurité](#)
  1. [12.4.1 Attaques d'usurpation de DNS](#)
  2. [12.4.2 Attaques inter-répertoires](#)
  3. [12.4.3 Risques de mise en œuvre](#)

## 12 Stockage Web



### 12.1 Présentation

*Cette section est non normative.*

Cette spécification introduit deux mécanismes connexes, similaires aux cookies de session HTTP, pour stocker les paires nom-valeur côté client. [\[BISCUITS\]](#)

Le premier est conçu pour les scénarios où l'utilisateur effectue une seule transaction, mais pourrait effectuer plusieurs transactions dans différentes fenêtres en même temps.

Les cookies ne gèrent pas vraiment bien ce cas. Par exemple, un utilisateur peut acheter des billets d'avion dans deux fenêtres différentes, en utilisant le même site. Si le site utilisait des cookies pour garder une trace du billet acheté par l'utilisateur, alors lorsque l'utilisateur cliquait d'une page à l'autre dans les deux

fenêtres, le billet en cours d'achat "fuyait" d'une fenêtre à l'autre, ce qui pouvait amener l'utilisateur à acheter deux billets pour le même vol sans s'en apercevoir.

Pour résoudre ce problème, cette spécification introduit le `sessionStorage` getter. Les sites peuvent ajouter des données au stockage de session, et il sera accessible à n'importe quelle page du même site ouvert dans cette fenêtre.

Par exemple, une page peut avoir une case à cocher que l'utilisateur coche pour indiquer qu'il souhaite une assurance :

```
<label>
  <input type="checkbox" onchange="sessionStorage.insurance =
checked ? 'true' : ''">
  I want insurance on this trip.
</label>
```

Une page ultérieure pourrait alors vérifier, à partir du script, si l'utilisateur a coché la case ou non :

```
if (sessionStorage.insurance) { ... }
```

Si l'utilisateur avait plusieurs fenêtres ouvertes sur le site, chacune aurait sa propre copie individuelle de l'objet de stockage de session.

Le deuxième mécanisme de stockage est conçu pour le stockage qui s'étend sur plusieurs fenêtres et dure au-delà de la session en cours. En particulier, les applications Web peuvent souhaiter stocker des mégaoctets de données utilisateur, telles que des documents entiers créés par l'utilisateur ou la boîte aux lettres d'un utilisateur, côté client pour des raisons de performances.

Encore une fois, les cookies ne gèrent pas bien ce cas, car ils sont transmis à chaque demande.

Le `localStorage` getter est utilisé pour accéder à la zone de stockage local d'une page.

Le site `example.com` peut afficher le nombre de fois que l'utilisateur a chargé sa page en mettant ce qui suit au bas de sa page :

```
<p>
  You have viewed this page
  <span id="count">an untold number of</span>
  time(s) .
</p>
<script>
  if (!localStorage.pageLoadCount)
    localStorage.pageLoadCount = 0;
```

```

    localStorage.pageLoadCount = parseInt(localStorage.pageLoadCount)
+ 1;

    document.getElementById('count').textContent =
localStorage.pageLoadCount;
</script>

```

Chaque site a sa propre zone de stockage séparée.

**Le `localStorage` getter donne accès à l'état partagé. Cette spécification ne définit pas l'interaction avec d'autres grappes d'agents dans un agent utilisateur multiprocessus, et les auteurs sont encouragés à supposer qu'il n'y a pas de mécanisme de verrouillage. Un site pourrait, par exemple, essayer de lire la valeur d'une clé, incrémenter sa valeur, puis la réécrire, en utilisant la nouvelle valeur comme identifiant unique pour la session ; si le site le fait deux fois dans deux fenêtres de navigateur différentes en même temps, il peut finir par utiliser le même identifiant "unique" pour les deux sessions, avec des effets potentiellement désastreux.**

## 12.2 L'API



### 12.2.1 L' `Storage` interface

```

[Exposed=Window]

interface Storage {

    readonly attribute unsigned long length;

    DOMString? key(unsigned long index);

    getter DOMString? getItem(DOMString key);

    setter undefined setItem(DOMString key, DOMString value);

    deleter undefined removeItem(DOMString key);

    undefined clear();

};

```

`storage.length`

✓

Renvoie le nombre de paires clé/valeur.

`storage.key (n)`

✓

Renvoie le nom de la *n* ième clé, ou null si *n* est supérieur ou égal au nombre de paires clé/valeur.

`value = storage.getItem (key)`

✓

`value = storage[key]`

Renvoie la valeur actuelle associée à la *clé* donnée , ou null si la *clé* donnée n'existe pas.

`storage.setItem (key, value)`

✓

`storage[key] = value`

Définit la valeur de la paire identifiée par *key* sur *value* , en créant une nouvelle paire clé/valeur si aucune n'existait auparavant pour *key* .

Lance une exception "[QuotaExceededError](#)" [DOMException](#) si la nouvelle valeur n'a pas pu être définie. (Le réglage peut échouer si, par exemple, l'utilisateur a désactivé le stockage pour le site ou si le quota a été dépassé.)

Distribue un [storage](#)événement sur [Window](#)les objets contenant un [Storage](#)objet équivalent.

`storage.removeItem (key)`

✓

`delete stockage [ clé ]`

Supprime la paire clé/valeur avec la *clé* donnée , si une paire clé/valeur avec la *clé* donnée existe.

Distribue un [storage](#)événement sur [Window](#)les objets contenant un [Storage](#)objet équivalent.

`storage.clear()`

✓

Supprime toutes les paires clé/valeur, s'il y en a.

Distribue un [storage](#)événement sur [Window](#)les objets contenant un [Storage](#)objet équivalent.

Un [Storage](#)objet est associé :

**carte**

Une [carte proxy de stockage](#) .

**taper**

" local " ou " session ".

Pour **réorganiser** un *stockage*[Storage](#) d'objets , réorganisez les [entrées](#) de la [carte](#) de *stockage* d'une manière [définie par l'implémentation](#) .

*Aussi malheureux que cela puisse paraître, l'ordre des itérations n'est pas défini et peut changer lors de la plupart des mutations.*

Pour **diffuser** un [Storage](#) objet *storage* , étant donné une *clé* , *oldValue* et *newValue* , exécutez ces étapes :

1. Soit *thisDocument* l' objet [global pertinent](#) du *stockage* [associé](#) .*Document*
2. Soit *url* l' [URL](#) de ce document .
3. Soit *remoteStorages* tous [Storage](#) les objets à l'exception du *stockage* dont :
  - o [type](#) est [le type](#) de *stockage*
  - o [l'origine](#) de [l'objet de paramètres pertinent](#) est [la même origine](#) que [l'origine](#) de l' objet [de paramètres pertinents](#) du *stockage* .

et, si [type](#) est " session ", dont [le nœud navigable](#) du nœud [navigable](#) associé [à l'objet de paramètres pertinent](#) est le [navigable traversable](#) du [nœud](#) [navigable](#) de ce *document* .*Document*

4. [Pour chaque](#) *remoteStorage* de *remoteStorages* : [mettez en file d'attente une tâche globale](#) sur la [source de la tâche de manipulation DOM](#) en fonction de [l'objet global pertinent](#) de *remoteStorage* pour [déclencher un événement](#) nommé sur [l'objet global pertinent](#) de *remoteStorage* , en utilisant , avec initialisé à *la clé* , initialisé à *oldValue* , initialisé à *newValue* , initialisé à *url* et initialisé à *remoteStorage* .[storageStorageEventkeyoldValuenewValueurlstorageArea](#)

*L' [Document](#) objet associé à la [tâche](#) résultante n'est pas nécessairement [entièrement actif](#) , mais les événements déclenchés sur ces objets sont ignorés par la [boucle d'événements](#) jusqu'à ce que [Document](#) redevienne [entièrement actif](#) .*

---

Les *length* étapes du getter consistent à retourner la [taille](#) de [cette carte](#) .

Les étapes de la méthode sont : *key (index)*

1. Si *index* est supérieur ou égal à la [taille](#) de [cette carte](#) , alors renvoie null.



2. Soit *keys* le résultat de l'exécution de `get` [the keys](#) sur [cette](#) carte .
3. *Touches* de retour [ *index* ].

Les [noms de propriété pris en charge](#) sur un *stockage*[Storage](#) d'objets sont le résultat de l'exécution [de get the keys](#) sur la [carte](#) du *stockage* .

Les étapes de la méthode sont : `getItem(key)`

1. Si [cette](#) carte [ *clé* ] n'existe pas , alors renvoyez null.
2. Retourne [cette](#) carte [ *clé* ] .

La méthode est : `setItem(key, value)`

1. Laissez *oldValue* être nul.
2. Soit *la réorganisation* vraie.
3. Si [cette carte](#) [ *clé* ] [existe](#) : \_
  1. Définissez *oldValue* sur [cette](#) carte [ *clé* ] .
  2. Si *oldValue* [est](#) *value* , alors retournez.
  3. Définissez *la réorganisation* sur faux.
4. Si *la valeur* ne peut pas être stockée, lancez une exception `"QuotaExceededError".DOMException`
5. [Définissez cette](#) carte [ *clé* ] sur *valeur* .
6. Si *reorder* est vrai, alors [reorder this](#) .
7. [Diffusez ceci](#) avec *key* , *oldValue* et *value* .

Les étapes de la méthode sont : `removeItem(key)`

1. Si [cette](#) carte [ *clé* ] n'existe pas , alors renvoyez null.
2. Définissez *oldValue* sur [cette](#) carte [ *clé* ] .
3. [Supprimer cette](#) carte [ *clé* ] .
4. [Recommandez ceci](#) .
5. [Diffusez ceci](#) avec *key* , *oldValue* et null.

Les `clear()` étapes de la méthode sont :

1. [Effacer cette](#) carte . \_

2. [Diffusez ceci](#) avec null, null et null.

### 12.2.2 Le [sessionStorage](#)getter

```
interface mixin WindowSessionStorage {
```

```
  readonly attribute Storage sessionStorage;
```

```
};
```

```
Window includes WindowSessionStorage;
```

**[window.sessionStorage](#)**

✓

Renvoie l' [Storage](#) objet associé à la zone de stockage de session de l'origine de cette *fenêtre* .

Lève un "[\\_SecurityError\\_](#) [DOMException](#)" si l' [origineDocument](#) de est une [origine opaque](#) ou si la demande viole une décision de politique (par exemple, si l'agent utilisateur est configuré pour ne pas autoriser la page à conserver les données).

Un [Document](#) objet a un **détenteur de stockage de session** associé , qui est null ou un [Storage](#) objet. Il est initialement nul.

Les [sessionStorage](#) étapes du getter sont :

1. Si [le détenteur de stockage](#) de session de [cet associé](#) [Document](#) n'est pas nul, alors renvoie le [détenteur de stockage](#) de session de [cet associé](#) [Document](#)
2. Soit *map* le résultat de l'exécution de l' [obtention d'un mappage de bouteille de stockage de session](#) avec [cet objet](#) de paramètres pertinent et "[sessionStorage](#)".
3. Si *la carte* est un échec, lancez un "[\\_SecurityError\\_](#) [DOMException](#)" .
4. Soit *storage* un nouvel [Storage](#) objet dont [map](#) est *map* .
5. Définissez [le détenteur de stockage](#) de session de [cet associé](#) [Document](#) sur *storage* .
6. *Stockage* de retour .

Après avoir créé un nouveau contexte de navigation auxiliaire et document , le stockage de la session est copié .

### 12.2.3 Le localStoragegetter

```
interface mixin WindowLocalStorage {
```

```
  readonly attribute Storage localStorage;
```

```
};
```

```
Window includes WindowLocalStorage;
```

**window.localStorage**

✓

Renvoie l' Storageobjet associé à la zone de stockage local de l'origine de la *fenêtre* .

Lève un "SecurityError" DOMException si l' origineDocument de est une origine opaque ou si la demande viole une décision de politique (par exemple, si l'agent utilisateur est configuré pour ne pas autoriser la page à conserver les données).

Un Documentobjet a un **support de stockage local** associé , qui est null ou un Storageobjet. Il est initialement nul.

Les localStorageétapes du getter sont :

1. Si le détenteur de stockage local de cet associé n'est Document pas nul, alors renvoie le détenteur de stockage local de cet associé .Document
2. Soit *map* le résultat de l'exécution de l'obtention d'un mappage de bouteille de stockage local avec cet objet de paramètres pertinent et "localStorage".
3. Si *la carte* est un échec, lancez un "SecurityError" DOMException .
4. Soit *storage* un nouvel Storageobjet dont map est *map* .
5. Définissez le support de stockage local de cet associé Document sur *storage* .
6. *Stockage* de retour .

## 12.2.4 L' StorageEvent interface



[Exposed=Window]

interface **StorageEvent** : [Event](#) {

    constructor(DOMString type, optional [StorageEventInit](#)

    eventInitDict = {});

    readonly attribute DOMString? [key](#);

    readonly attribute DOMString? [oldValue](#);

    readonly attribute DOMString? [newValue](#);

    readonly attribute USVString [url](#);

    readonly attribute [Storage](#)? [storageArea](#);

    undefined [initStorageEvent](#)(DOMString type, optional boolean

    bubbles = false, optional boolean cancelable = false, optional

    DOMString? key = null, optional DOMString? oldValue = null,

    optional DOMString? newValue = null, optional USVString url = "",

    optional Storage? storageArea = null);

};

dictionary **StorageEventInit** : [EventInit](#) {

    DOMString? key = null;

    DOMString? oldValue = null;

    DOMString? newValue = null;

    USVString url = "";

```
Storage? storageArea = null;
```

```
};
```

**event.key**

Renvoie la clé de l'élément de stockage en cours de modification.

**event.oldValue**

Renvoie l'ancienne valeur de la clé de l'élément de stockage dont la valeur est modifiée.

**event.newValue**

Renvoie la nouvelle valeur de la clé de l'élément de stockage dont la valeur est modifiée.

**event.url**

Renvoie l' [URL](#) du document dont l'élément de stockage a changé.

**event.storageArea**

Renvoie l' [Storage](#) objet qui a été affecté.

Les attributs **key**, **oldValue**, , et doivent renvoyer les valeurs avec **newValue** lesquelles ils ont été initialisés. **url** **storageArea**

La méthode doit initialiser l'événement d'une manière analogue à la méthode portant le même

nom. [DOM](#) **initStorageEvent**(*type*, *bubbles*, *cancelable*, *key*, *oldValue*, *newValue*, *url*, *storageArea*) **initEvent**()

## 12.3 Confidentialité

### 12.3.1 Suivi des utilisateurs

Un annonceur tiers (ou toute entité capable de diffuser du contenu sur plusieurs sites) pourrait utiliser un identifiant unique stocké dans sa zone de stockage locale pour suivre un utilisateur sur plusieurs sessions, en créant un profil des intérêts de l'utilisateur pour permettre une publicité très ciblée. . En conjonction avec un site qui connaît l'identité réelle de l'utilisateur (par exemple un site de commerce électronique qui nécessite des informations d'identification authentifiées), cela pourrait permettre à des groupes oppressifs de cibler des individus avec une plus grande précision que dans un monde où l'utilisation du Web est purement anonyme.

Il existe un certain nombre de techniques qui peuvent être utilisées pour atténuer le risque de suivi des utilisateurs :

#### Blocage du stockage tiers

Les agents utilisateurs peuvent restreindre l'accès aux [localStorage](#) objets aux scripts provenant du domaine du [document actif](#) du [traversable de niveau supérieur](#), par exemple en refusant l'accès à l'API pour les pages d'autres domaines s'exécutant dans [iframes](#).

### **Expiration des données stockées**

Les agents utilisateurs peuvent, éventuellement d'une manière configurée par l'utilisateur, supprimer automatiquement les données stockées après un certain temps.

Par exemple, un agent utilisateur pourrait être configuré pour traiter les zones de stockage locales tierces comme un stockage de session uniquement, en supprimant les données une fois que l'utilisateur a fermé tous les [navigables](#) qui pourraient y accéder.

Cela peut limiter la capacité d'un site à suivre un utilisateur, car le site ne pourra alors suivre l'utilisateur sur plusieurs sessions que lorsqu'il s'authentifiera sur le site lui-même (par exemple en effectuant un achat ou en se connectant à un service).

Cependant, cela réduit également l'utilité de l'API en tant que mécanisme de stockage à long terme. Cela peut également mettre les données de l'utilisateur en danger, si l'utilisateur ne comprend pas pleinement les implications de l'expiration des données.

### **Traiter le stockage persistant comme des cookies**

Si les utilisateurs tentent de protéger leur vie privée en effaçant les cookies sans effacer également les données stockées dans la zone de stockage locale, les sites peuvent déjouer ces tentatives en utilisant les deux fonctionnalités comme sauvegarde redondante l'une pour l'autre. Les agents utilisateurs devraient présenter les interfaces pour les effacer d'une manière qui aide les utilisateurs à comprendre cette possibilité et leur permette de supprimer simultanément des données dans toutes les fonctionnalités de stockage persistant. [\[BISCUITS\]](#)

### **Liste sécurisée spécifique au site de l'accès aux zones de stockage locales**

Les agents utilisateurs peuvent autoriser les sites à accéder aux zones de stockage de session de manière illimitée, mais exigent que l'utilisateur autorise l'accès aux zones de stockage locales.

### **Suivi de l'origine des données stockées**

Les agents utilisateurs peuvent enregistrer les [origines](#) des sites qui contenaient du contenu d'origines tierces qui ont entraîné le stockage des données.

Si ces informations sont ensuite utilisées pour présenter la vue des données actuellement dans le stockage persistant, cela permettrait à l'utilisateur de prendre des décisions éclairées sur les parties du stockage persistant à élaguer. Combiné avec une liste de blocage ("supprimez ces données et

empêchez ce domaine de stocker à nouveau des données"), l'utilisateur peut restreindre l'utilisation du stockage persistant aux sites auxquels il fait confiance.

### **Listes de blocage partagées**

Les agents utilisateurs peuvent autoriser les utilisateurs à partager leurs listes de blocage de domaine de stockage persistant.

Cela permettrait aux communautés d'agir ensemble pour protéger leur vie privée.

Bien que ces suggestions empêchent une utilisation triviale de cette API pour le suivi des utilisateurs, elles ne la bloquent pas complètement. Au sein d'un même domaine, un site peut continuer à suivre l'utilisateur pendant une session, puis transmettre toutes ces informations au tiers ainsi que toute information d'identification (noms, numéros de carte de crédit, adresses) obtenue par le site. Si un tiers coopère avec plusieurs sites pour obtenir de telles informations, un profil peut toujours être créé.

Cependant, le suivi des utilisateurs est dans une certaine mesure possible même sans aucune coopération de la part de l'agent utilisateur, par exemple en utilisant des identifiants de session dans les URL, une technique déjà couramment utilisée à des fins anodines mais facilement réutilisée pour le suivi des utilisateurs (même rétroactivement). Ces informations peuvent ensuite être partagées avec d'autres sites, en utilisant les adresses IP des visiteurs et d'autres données spécifiques à l'utilisateur (par exemple, les en-têtes d'agent utilisateur et les paramètres de configuration) pour combiner des sessions distinctes dans des profils utilisateur cohérents.

### **12.3.2 Sensibilité des données**

Les agents utilisateurs doivent traiter les données stockées de manière persistante comme potentiellement sensibles ; il est tout à fait possible que des e-mails, des rendez-vous du calendrier, des dossiers de santé ou d'autres documents confidentiels soient stockés dans ce mécanisme.

À cette fin, les agents utilisateurs doivent s'assurer que lors de la suppression de données, elles sont rapidement supprimées du stockage sous-jacent.

## **12.4 Sécurité**

### **12.4.1 Attaques d'usurpation de DNS**

En raison du potentiel d'attaques d'usurpation de DNS, on ne peut pas garantir qu'un hôte prétendant appartenir à un certain domaine appartient réellement à ce domaine. Pour atténuer cela, les pages peuvent utiliser TLS. Les pages utilisant TLS peuvent être sûres que seuls l'utilisateur, le logiciel travaillant pour le compte de l'utilisateur et les autres pages utilisant TLS qui ont des certificats les identifiant comme appartenant au même domaine peuvent accéder à leurs zones de stockage.

### 12.4.2 Attaques inter-répertoires

Différents auteurs partageant un même nom d'hôte, par exemple les utilisateurs hébergeant du contenu sur le défunt `geocities.com`, partagent tous un objet de stockage local. Il n'y a pas de fonctionnalité pour restreindre l'accès par nom de chemin. Les auteurs sur des hébergeurs partagés sont donc instamment priés d'éviter d'utiliser ces fonctionnalités, car il serait trivial pour d'autres auteurs de lire les données et de les écraser.

*Même si une fonctionnalité de restriction de chemin était rendue disponible, le modèle de sécurité de script DOM habituel rendrait trivial le contournement de cette protection et l'accès aux données depuis n'importe quel chemin.*

### 12.4.3 Risques de mise en œuvre

Les deux principaux risques lors de la mise en œuvre de ces fonctionnalités de stockage persistant sont de laisser les sites hostiles lire des informations à partir d'autres domaines et de laisser les sites hostiles écrire des informations qui sont ensuite lues à partir d'autres domaines.

Laisser des sites tiers lire des données qui ne sont pas censées être lues à partir de leur domaine entraîne *une fuite d'informations*. Par exemple, la liste de souhaits d'achat d'un utilisateur sur un domaine peut être utilisée par un autre domaine pour une publicité ciblée ; ou les documents confidentiels en cours d'élaboration d'un utilisateur stockés par un site de traitement de texte pourraient être examinés par le site d'une société concurrente.

Laisser des sites tiers écrire des données dans le stockage persistant d'autres domaines peut entraîner une *usurpation d'informations*, ce qui est tout aussi dangereux. Par exemple, un site hostile pourrait ajouter des éléments à la liste de souhaits d'un utilisateur ; ou un site hostile peut définir l'identifiant de session d'un utilisateur sur un identifiant connu que le site hostile peut ensuite utiliser pour suivre les actions de l'utilisateur sur le site victime.

Ainsi, suivre strictement le modèle [d'origine](#) décrit dans cette spécification est important pour la sécurité de l'utilisateur.



## 1. [13 La syntaxe HTML](#)

### 1. [13.1 Écrire des documents HTML](#)

1. [13.1.1 Le DOCTYPE](#)
2. [13.1.2 Éléments](#)
  1. [13.1.2.1 Balises de début](#)
  2. [13.1.2.2 Balises de fin](#)
  3. [13.1.2.3 Attributs](#)
  4. [13.1.2.4 Balises facultatives](#)
  5. [13.1.2.5 Restrictions sur les modèles de contenu](#)
  6. [13.1.2.6 Restrictions sur le contenu du texte brut et des éléments de texte brut échappables](#)
3. [13.1.3 Texte](#)
  1. [13.1.3.1 Nouvelles lignes](#)
4. [13.1.4 Références de caractères](#)
5. [13.1.5 Rubriques CDATA](#)
6. [13.1.6 Commentaires](#)

## 13 La syntaxe HTML

*Cette section décrit uniquement les règles pour les ressources étiquetées avec un [type HTML MIME](#) . Les règles pour les ressources XML sont abordées dans la section ci-dessous intitulée " [La syntaxe XML](#) " .*

### 13.1 Écrire des documents HTML

*Cette section s'applique uniquement aux documents, aux outils de création et aux générateurs de balisage. En particulier, elle ne s'applique pas aux vérificateurs de conformité ; les vérificateurs de conformité doivent utiliser les exigences indiquées dans la section suivante ("analyse des documents HTML").*

Les documents doivent comprendre les parties suivantes, dans l'ordre indiqué :

1. Facultativement, un seul caractère U+FEFF BYTE ORDER MARK (BOM).
2. N'importe quel nombre de [commentaires](#) et [d'espaces ASCII](#) .
3. UN [DOCTYPE](#) .

4. N'importe quel nombre de [commentaires](#) et [d'espaces ASCII](#) .
5. L' [élément document](#) , sous la forme d'un [html élément](#) .
6. N'importe quel nombre de [commentaires](#) et [d'espaces ASCII](#) .

Les différents types de contenu mentionnés ci-dessus sont décrits dans les quelques sections suivantes.

De plus, il existe certaines restrictions sur la manière dont [les déclarations de codage de caractères](#) doivent être sérialisées, comme indiqué dans la section sur ce sujet.

*[Les espaces blancs ASCII](#) avant l' [html élément](#), au début de l' [html élément](#) et avant l' [head élément](#), seront supprimés lors de l'analyse du document ; [Les espaces blancs ASCII](#) après l' [html élément](#) seront analysés comme s'ils se trouvaient à la fin de l' [body élément](#). Ainsi, [les espaces blancs ASCII](#) autour de l' [élément de document](#) ne font pas d'aller-retour.*

*Il est suggéré d'insérer des sauts de ligne après le DOCTYPE, après tout commentaire avant l'élément de document, après la [html](#) balise de début de l'élément (si elle n'est pas [omise](#) ) et après tout commentaire à l'intérieur de l' [html élément](#) mais avant l' [head élément](#).*

De nombreuses chaînes de la syntaxe HTML (par exemple, les noms des éléments et leurs attributs) ne sont pas sensibles à la casse, mais uniquement pour [les alphas supérieurs ASCII](#) et [les alphas inférieurs ASCII](#) . Pour plus de commodité, dans cette section, cela est simplement appelé "insensible à la casse".

### 13.1.1 Le DOCTYPE

Un **DOCTYPE** est un préambule obligatoire.

*Les DOCTYPEs sont requis pour des raisons d'héritage. Lorsqu'il est omis, les navigateurs ont tendance à utiliser un mode de rendu différent qui est incompatible avec certaines spécifications. L'inclusion du DOCTYPE dans un document garantit que le navigateur fait de son mieux pour suivre les spécifications pertinentes.*

Un DOCTYPE doit être composé des composants suivants, dans cet ordre :

1. Une chaîne qui est une correspondance [ASCII non sensible à la casse](#) pour la chaîne "`<!DOCTYPE`".
2. Un ou plusieurs [espaces blancs ASCII](#) .
3. Une chaîne qui est une correspondance [ASCII non sensible à la casse](#) pour la chaîne "`html`".
4. Facultativement, une [ancienne chaîne DOCTYPE](#) .
5. Zéro ou plusieurs [espaces blancs ASCII](#) .

6. Un caractère U+003E SIGNE SUPÉRIEUR À (>).

*En d'autres termes, <!DOCTYPE html>, insensible à la casse.*

---

Pour les besoins des générateurs HTML qui ne peuvent pas produire de balisage HTML avec le court DOCTYPE "`<!DOCTYPE html>`", une **chaîne héritée DOCTYPE** peut être insérée dans le DOCTYPE (dans la position définie ci-dessus). Cette chaîne doit être composée de :

1. Un ou plusieurs [espaces blancs ASCII](#) .
2. Une chaîne qui est une correspondance [ASCII non sensible à la casse](#) pour la chaîne "`SYSTEM`".
3. Un ou plusieurs [espaces blancs ASCII](#) .
4. Un caractère U+0022 GUILLEMET ou U+0027 APOSTROPHE (le *guillemet* ).
5. La chaîne littérale "`about:legacy-compat`".
6. Un caractère U+0022 QUOTATION MARK ou U+0027 APOSTROPHE correspondant (c'est-à-dire le même caractère qu'à l'étape précédente étiqueté *guillemet* ).

*En d'autres termes, <!DOCTYPE html SYSTEM "about:legacy-compat">OU <!DOCTYPE html SYSTEM 'about:legacy-compat'>, insensible à la casse sauf pour la partie entre guillemets simples ou doubles.*

La [chaîne héritée DOCTYPE](#) ne doit pas être utilisée à moins que le document ne soit généré à partir d'un système qui ne peut pas produire la chaîne la plus courte.

### 13.1.2 Éléments

Il existe six types d' **éléments** différents : [les éléments vides](#) , l' [template élément](#) , [les éléments de texte brut](#) , [les éléments de texte brut échappables](#) , [les éléments étrangers](#) et [les éléments normaux](#) .

#### **Éléments vides**

[area](#), [base](#), [br](#), [col](#), [embed](#), [hr](#), [img](#), [input](#), [link](#), [meta](#), [source](#), [track](#), [wbr](#)

#### **L' [template](#) élément**

[template](#)

#### **Éléments de texte brut**

[script](#), [style](#)

#### **Éléments de texte brut échappables**

[textarea](#), [title](#)

## Éléments étrangers

Éléments de l' [espace de noms MathML](#) et de l' [espace de noms SVG](#) .

## Éléments normaux

[Tous les autres éléments HTML](#) autorisés sont des éléments normaux.

**Les balises** sont utilisées pour délimiter le début et la fin des éléments dans le balisage. [Le texte brut](#) , [le texte brut échappable](#) et les éléments [normaux](#) ont une [balise de début](#) pour indiquer où ils commencent et une [balise de fin](#) pour indiquer où ils se terminent. Les balises de début et de fin de certains [éléments normaux](#) peuvent être [omis](#) , comme décrit ci-dessous dans la section sur [les balises facultatives](#) . Ceux qui ne peuvent pas être omis ne doivent pas être omis. [Les éléments vides](#) n'ont qu'une balise de début ; Les balises de fin ne doivent pas être spécifiées pour [les éléments vides](#) . [Éléments étrangers](#) doivent soit avoir une balise de début et une balise de fin, soit une balise de début marquée comme se fermant automatiquement, auquel cas elles ne doivent pas avoir de balise de fin.

Le [contenu](#) de l'élément doit être placé entre juste après la balise de début (qui [peut être implicite, dans certains cas](#) ) et juste avant la balise de fin (qui, encore une fois, [peut être implicite dans certains cas](#) ). Le contenu exact autorisé de chaque élément individuel dépend du [modèle de contenu](#) de cet élément, comme décrit précédemment dans cette spécification. Les éléments ne doivent pas contenir de contenu que leur modèle de contenu interdit. En plus des restrictions imposées au contenu par ces modèles de contenu, cependant, les cinq types d'éléments ont des exigences *syntactiques* supplémentaires .

[Les éléments vides](#) ne peuvent pas avoir de contenu (puisque'il n'y a pas de balise de fin, aucun contenu ne peut être placé entre la balise de début et la balise de fin).

L' [template](#)élément peut avoir [un contenu de modèle](#) , mais ce [contenu de modèle](#) n'est pas un enfant de l' [template](#)élément lui-même. Au lieu de cela, ils sont stockés dans un [DocumentFragment](#) associé à un autre [Document](#)- sans [contexte de navigation](#) - afin d'éviter que le [template](#)contenu n'interfère avec le principal [Document](#). Le balisage du [contenu du modèle](#) d'un [template](#)élément est placé juste après la [template](#)balise de début de l'élément et juste avant [template](#) la balise de fin de l'élément (comme pour les autres éléments), et peut consister en n'importe quel [texte](#) , [références de caractères](#) , [éléments](#) et [commentaires](#), mais le texte ne doit pas contenir le caractère U+003C SIGNE INFÉRIEUR À (<) ou une [esperluette ambiguë](#) .

[Les éléments de texte brut](#) peuvent avoir [du texte](#) , bien qu'il ait [des restrictions](#) décrites ci-dessous.

[Les éléments de texte brut échappables](#) peuvent avoir [des références de texte](#) et de caractère , mais le texte ne doit pas contenir d' [esperluette ambiguë](#) . Il existe également [d'autres restrictions](#) décrites ci-dessous.

[Les éléments étrangers](#) dont la balise de début est marquée comme se fermant automatiquement ne peuvent pas avoir de contenu (puisque, encore une fois,

comme il n'y a pas de balise de fin, aucun contenu ne peut être placé entre la balise de début et la balise de fin). [Les éléments étrangers](#) dont la balise de début *n'est pas* marquée comme à fermeture automatique peuvent avoir du [texte](#) , [des références de caractères](#) , [des sections CDATA](#) , d'autres [éléments](#) et [des commentaires](#) , mais le texte ne doit pas contenir le caractère U+003C SIGNE MOINS QUE (<) ou une [esperluette ambiguë](#) .

*La syntaxe HTML ne prend pas en charge les déclarations d'espace de noms, même dans [les éléments étrangers](#) .*

*Par exemple, considérez le fragment HTML suivant :*

```
<p>
  <svg>
    <metadata>
      <!-- this is invalid -->
      <cdr:license xmlns:cdr="https://www.example.com/cdr/metadata"
name="MIT"/>
    </metadata>
  </svg>
</p>
```

*L'élément le plus interne, `cdr:license`, se trouve en fait dans l'espace de noms SVG, car l'`xmlns:cdr` attribut " " n'a aucun effet (contrairement à XML). En fait, comme le dit le commentaire dans le fragment ci-dessus, le fragment est en fait non conforme. C'est parce que SVG 2 ne définit aucun élément appelé " `cdr:license` " dans l'espace de noms SVG.*

[Les éléments normaux](#) peuvent avoir [du texte](#) , [des références de caractères](#) , d'autres [éléments](#) et [des commentaires](#) , mais le texte ne doit pas contenir le caractère U+003C SIGNE MOINS-QUE (<) ou une [esperluette ambiguë](#) . Certains [éléments normaux](#) ont également [encore plus de restrictions](#) sur le contenu qu'ils sont autorisés à contenir, au-delà des restrictions imposées par le modèle de contenu et celles décrites dans ce paragraphe. Ces restrictions sont décrites ci-dessous.

Les balises contiennent un **nom de balise** , donnant le nom de l'élément. Les éléments HTML ont tous des noms qui utilisent uniquement [des caractères alphanumériques ASCII](#) . Dans la syntaxe HTML, les noms de balises, même ceux des [éléments étrangers](#) , peuvent être écrits avec n'importe quelle combinaison de lettres minuscules et majuscules qui, une fois converties en minuscules, correspondent au nom de balise de l'élément ; les noms de balises ne sont pas sensibles à la casse.

### 13.1.2.1 Balises de début

**Les balises de début** doivent avoir le format suivant :

1. Le premier caractère d'une balise de début doit être un caractère U+003C SIGNE MOINS QUE (<).
2. Les quelques caractères suivants d'une balise de début doivent être le [nom de la balise](#) de l'élément .
3. S'il doit y avoir des attributs à l'étape suivante, il doit d'abord y avoir un ou plusieurs [espaces blancs ASCII](#) .
4. Ensuite, la balise de début peut avoir un certain nombre d'attributs, [dont la syntaxe](#) est décrite ci-dessous. Les attributs doivent être séparés les uns des autres par un ou plusieurs [espaces blancs ASCII](#) .
5. Après les attributs, ou après le [nom de la balise](#) s'il n'y a pas d'attributs, il peut y avoir un ou plusieurs [espaces blancs ASCII](#) . (Certains attributs doivent être suivis d'un espace. Voir la [section des attributs](#) ci-dessous.)
6. Ensuite, si l'élément est l'un des [éléments vides](#) , ou si l'élément est un [élément étranger](#) , il peut y avoir un seul caractère U+002F SOLIDUS (/), qui sur [les éléments étrangers](#) marque la balise de début comme se fermant automatiquement. Sur [les éléments void](#) , il ne marque pas la balise de début comme se fermant automatiquement, mais est inutile et n'a aucun effet d'aucune sorte. Pour de tels éléments vides, il ne doit être utilisé qu'avec prudence - d'autant plus que, s'il est directement précédé d'une [valeur d'attribut sans guillemets](#) , il fait partie de la valeur d'attribut plutôt que d'être rejeté par l'analyseur.
7. Enfin, les balises de début doivent être fermées par un caractère U+003E SIGNE SUPÉRIEUR À (>).

### 13.1.2.2 Balises de fin

**Les balises de fin** doivent avoir le format suivant :

1. Le premier caractère d'une balise de fin doit être un caractère SIGNE INFÉRIEUR À U+003C (<).
2. Le deuxième caractère d'une balise de fin doit être un caractère U+002F SOLIDUS (/).
3. Les quelques caractères suivants d'une balise de fin doivent être le [nom de la balise](#) de l'élément .
4. Après le nom de la balise, il peut y avoir un ou plusieurs [espaces blancs ASCII](#) .

5. Enfin, les balises de fin doivent être fermées par un caractère U+003E SIGNE SUPÉRIEUR À (>).

### 13.1.2.3 Attributs

**Les attributs** d'un élément sont exprimés à l'intérieur de la balise de début de l'élément.

Les attributs ont un nom et une valeur. **Les noms d'attribut** doivent être composés d'un ou plusieurs caractères autres que [les contrôles](#) , U+0020 ESPACE, U+0022 ("), U+0027 ('), U+003E (>), U+002F (/), U+003D (=) et [des non-caractères](#) . Dans la syntaxe HTML, les noms d'attributs, même ceux des [éléments étrangers](#) , peuvent être écrits avec n'importe quelle combinaison d' alphas [ASCII inférieurs](#) et [ASCII supérieurs](#) .

**Les valeurs d'attribut** sont un mélange de [références de texte](#) et de caractères , sauf avec la restriction supplémentaire que le texte ne peut pas contenir d' [esperluette ambiguë](#) .

Les attributs peuvent être spécifiés de quatre manières différentes :

#### Syntaxe d'attribut vide

Juste le [nom de l'attribut](#) . La valeur est implicitement la chaîne vide.

Dans l'exemple suivant, l' [disabled](#) attribut est donné avec la syntaxe d'attribut vide :

```
<input disabled>
```

Si un attribut utilisant la syntaxe d'attribut vide doit être suivi d'un autre attribut, alors il doit y avoir [un espace blanc ASCII](#) séparant les deux.

#### Syntaxe de valeur d'attribut sans guillemets

Le [nom de l'attribut](#) , suivi de zéro ou plusieurs [espaces blancs ASCII](#) , suivis d'un seul caractère SIGNE ÉGAL U+003D , suivi de zéro ou plusieurs [espaces blancs ASCII](#) , suivis de la [valeur de l'attribut](#) , qui, en plus des exigences indiquées ci-dessus pour les valeurs d'attribut, ne doit pas contenir d' [espaces ASCII](#) littéraux, de caractères U+0022 GUILLEMET ("), U+0027 caractères APOSTROPHE ('), U+003D caractères SIGNE ÉGAL (=), U+003C caractères INFÉRIEUR À SIGNE (<), caractères U+003E SIGNE SUPÉRIEUR À (>) ou caractères U+0060 ACCENT GRAVE (`), et ne doit pas être la chaîne vide.

Dans l'exemple suivant, l' [value](#) attribut est donné avec la syntaxe de valeur d'attribut sans guillemets :

```
<input value=yes>
```



Si un attribut utilisant la syntaxe d'attribut sans guillemets doit être suivi d'un autre attribut ou du caractère optionnel U+002F SOLIDUS (/) autorisé à l'étape 6 de la syntaxe [de la balise de début](#) ci-dessus, il doit y avoir [un espace blanc ASCII](#) séparant les deux.

### Syntaxe de valeur d'attribut entre guillemets simples

Le [nom de l'attribut](#) , suivi de zéro ou plusieurs [espaces blancs ASCII](#) , suivis d'un seul caractère U+003D SIGNE ÉGAL , suivi de zéro ou plusieurs [espaces blancs ASCII](#) , suivis d'un seul caractère U+0027 APOSTROPHE ('), suivi de la [valeur de l'attribut](#) , qui, en plus des exigences indiquées ci-dessus pour les valeurs d'attribut, ne doit contenir aucun caractère littéral U+0027 APOSTROPHE ('), et enfin suivi d'un deuxième caractère unique U+0027 APOSTROPHE (').

Dans l'exemple suivant, l' [type](#) attribut est donné avec la syntaxe de valeur d'attribut entre guillemets simples :

```
<input type='checkbox'>
```

Si un attribut utilisant la syntaxe d'attribut entre guillemets simples doit être suivi d'un autre attribut, alors il doit y avoir un [espace blanc ASCII](#) séparant les deux.

### Syntaxe de valeur d'attribut entre guillemets doubles

Le [nom de l'attribut](#) , suivi de zéro ou plusieurs [espaces blancs ASCII](#) , suivis d'un seul caractère U+003D SIGNE ÉGAL, suivis de zéro ou plusieurs [espaces blancs ASCII](#) , suivis d'un seul caractère U+0022 QUOTATION MARK ("), suivis de la [valeur de l'attribut](#) , qui, en plus des exigences indiquées ci-dessus pour les valeurs d'attribut, ne doit contenir aucun caractère littéral U+0022 GUILLEMET ("), et enfin suivi d'un second caractère unique U+0022 GUILLEMET (").

Dans l'exemple suivant, l' [name](#) attribut est donné avec la syntaxe de valeur d'attribut entre guillemets :

```
<input name="be evil">
```

Si un attribut utilisant la syntaxe d'attribut entre guillemets doubles doit être suivi d'un autre attribut, alors il doit y avoir un [espace blanc ASCII](#) séparant les deux.

Il ne doit jamais y avoir deux ou plusieurs attributs sur la même balise ouvrante dont les noms sont une correspondance [ASCII insensible à la casse](#) l'un pour l'autre.

---



Lorsqu'un [élément étranger](#) a l'un des attributs d'espace de noms donnés par le nom local et l'espace de noms des première et deuxième cellules d'une ligne du tableau suivant, il doit être écrit en utilisant le nom donné par la troisième cellule de la même ligne.

Nom local	Espace de noms	Nom d'attribut
actuate	<a href="#">Espace de noms XLink</a>	xlink:actuate
arcrole	<a href="#">Espace de noms XLink</a>	xlink:arcrole
href	<a href="#">Espace de noms XLink</a>	xlink:href
role	<a href="#">Espace de noms XLink</a>	xlink:role
show	<a href="#">Espace de noms XLink</a>	xlink:show
title	<a href="#">Espace de noms XLink</a>	xlink:title
type	<a href="#">Espace de noms XLink</a>	xlink:type
lang	<a href="#">Espace de noms XML</a>	xml:lang
space	<a href="#">Espace de noms XML</a>	xml:space
xmlns	<a href="#">Espace de noms XMLNS</a>	xmlns
xlink	<a href="#">Espace de noms XMLNS</a>	xmlns:xlink

Aucun autre attribut d'espace de noms ne peut être exprimé dans [la syntaxe HTML](#) .

*Que les attributs du tableau ci-dessus soient conformes ou non est défini par d'autres spécifications (par exemple SVG 2 et MathML ) ; cette section décrit uniquement les règles de syntaxe si les attributs sont sérialisés à l'aide de la syntaxe HTML.*

#### 13.1.2.4 Balises facultatives

Certaines balises peuvent être **omises** .

*[L'omission de la balise de début](#) d'un élément dans les situations décrites ci-dessous ne signifie pas que l'élément n'est pas présent ; c'est sous-entendu, mais c'est toujours là. Par exemple, un document HTML a toujours un [html](#) élément racine, même si la chaîne `<html>` n'apparaît nulle part dans le balisage.*

[La balise de début](#) d'un [html](#) élément peut être omise si la première chose à l'intérieur de l' élément n'est pas un [commentaire](#) [.html](#)

Par exemple, dans le cas suivant, vous pouvez supprimer la `<html>` balise " " :

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>Hello</title>
```

```
</head>
<body>
  <p>Welcome to this example.</p>
</body>
</html>
```

Cela donnerait au document l'aspect suivant :

```
<!DOCTYPE HTML>

<head>
  <title>Hello</title>
</head>
<body>
  <p>Welcome to this example.</p>
</body>
</html>
```

Cela a exactement le même DOM. En particulier, notez que les espaces autour de l' [élément document](#) sont ignorés par l'analyseur. L'exemple suivant aurait également exactement le même DOM :

```
<!DOCTYPE HTML><head>
  <title>Hello</title>
</head>
<body>
  <p>Welcome to this example.</p>
</body>
</html>
```

Cependant, dans l'exemple suivant, la suppression de la balise de début déplace le commentaire avant l' [html](#) élément :

```
<!DOCTYPE HTML>
<html>
  <!-- where is this comment in the DOM? -->
  <head>
    <title>Hello</title>
  </head>
  <body>
    <p>Welcome to this example.</p>
  </body>
</html>
```

Avec la balise supprimée, le document devient en fait le même que celui-ci :

```
<!DOCTYPE HTML>
<!-- where is this comment in the DOM? -->
<html>
  <head>
    <title>Hello</title>
  </head>
  <body>
    <p>Welcome to this example.</p>
  </body>
</html>
```

C'est pourquoi la balise ne peut être supprimée que si elle n'est pas suivie d'un commentaire : la suppression de la balise lorsqu'il y a un commentaire modifie l'arbre d'analyse résultant du document. Bien sûr, si la position du commentaire n'a pas d'importance, la balise peut être omise, comme si le commentaire avait été déplacé avant la balise de début en premier lieu.

La [balise de fin](#) d'un [html](#) élément peut être omise si l'élément n'est pas immédiatement suivi d'un [commentaire](#) [.html](#)

La [balise de début](#)[head](#) d'un élément peut être omise si l'élément est vide ou si la première chose à l'intérieur de l'élément est un élément [.head](#)

La [balise de fin](#)[head](#) d'un élément peut être omise si l'élément n'est pas immédiatement suivi d'un [un espace ASCII](#) ou d'un [commentaire](#) [.head](#)

La [balise de début](#)[body](#) d'un élément peut être omise si l'élément est vide, ou si la première chose à l'intérieur de l'élément n'est pas [un espace ASCII](#) ou un [commentaire](#), sauf si la première chose à l'intérieur de l'élément est un élément [, , , ,](#) ou [.bodybodymetanascriptlinkscriptstyletemplate](#)

La [balise de fin](#)[body](#) d'un élément peut être omise si l'élément n'est pas immédiatement suivi d'un [commentaire](#) [.body](#)

Notez que dans l'exemple ci-dessus, les [head](#) balises de début et de fin d'élément, ainsi que la [body](#) balise de début d'élément, ne peuvent pas être omises, car elles sont entourées d'espace :

```
<!DOCTYPE HTML>
<html>
  <head>
    <title>Hello</title>
  </head>
  <body>
```

```
<p>Welcome to this example.</p>
</body>
</html>
```

(Les balises de fin d'élément `body` et `html` peuvent être omises sans problème ; tous les espaces après ceux-ci sont `body` de toute façon analysés dans l'élément.)

Habituellement, cependant, les espaces blancs ne sont pas un problème. Si nous supprimons d'abord l'espace blanc dont nous ne nous soucions pas :

```
<!DOCTYPE
HTML><html><head><title>Hello</title></head><body><p>Welcome to
this example.</p></body></html>
```

Ensuite, nous pouvons omettre un certain nombre de balises sans affecter le DOM :

```
<!DOCTYPE HTML><title>Hello</title><p>Welcome to this example.</p>
```

À ce stade, nous pouvons également rajouter des espaces :

```
<!DOCTYPE HTML>

<title>Hello</title>

<p>Welcome to this example.</p>
```

Cela équivaudrait à ce document, avec les balises omises affichées dans leurs positions implicites par l'analyseur ; le seul nœud de texte blanc qui en résulte est le retour à la ligne à la fin de l' `head` élément :

```
<!DOCTYPE HTML>

<html><head><title>Hello</title>

</head><body><p>Welcome to this example.</p></body></html>
```

La balise de fin d'un liélément peut être omise si l'élément est immédiatement suivi d'un autre élément ou s'il n'y a plus de contenu dans l'élément parent.lilili

La balise de fin<sup>dt</sup> d'un élément peut être omise si l'élément est immédiatement suivi d'un autre élément ou d'un élément.<sup>dt</sup><sup>dt</sup><sup>dd</sup>

La [balise de fin](#)`dd` d'un élément peut être omise si l'élément est immédiatement suivi d'un autre élément ou d'un élément, ou s'il n'y a plus de contenu dans l'élément parent.`ddddd`

La balise de fin `</>` d'un élément peut être omise si l'élément est immédiatement suivi d'un élément , , , , , , , , , , , , , , , , ou , ou s'il existe n'est plus contenu dans l'élément parent et l'élément parent est un [élément HTML](#) qui n'est pas un , , , , ,

ou un [élément](#), ou un [élément personnalisé autonome](#).

Nous pouvons ainsi simplifier davantage l'exemple précédent :

```
<!DOCTYPE HTML><title>Hello</title><p>Welcome to this example.
```

La [balise de fin](#) d'un [rt](#) élément peut être omise si l'élément est immédiatement suivi d'un élément ou , ou s'il n'y a plus de contenu dans l'élément parent.[rtrtrp](#)

La [balise de fin](#) d'un [rp](#) élément peut être omise si l'élément est immédiatement suivi d'un élément ou , ou s'il n'y a plus de contenu dans l'élément parent.[rprtrp](#)

La [balise de fin](#) d'un [optgroup](#) élément peut être omise si l'élément est immédiatement suivi d'un autre élément, ou s'il n'y a plus de contenu dans l'élément parent.[optgroupoptgroup](#)

La [balise de fin](#) d'un [option](#) élément peut être omise si l'élément est immédiatement suivi d'un autre élément, ou s'il est immédiatement suivi d'un élément, ou s'il n'y a plus de contenu dans l'élément parent.[optionoptionoptgroup](#)

La [balise de début](#)[colgroup](#) d'un élément peut être omise si la première chose à l'intérieur de l'élément est un élément, et si l'élément n'est pas immédiatement précédé d'un autre élément dont [la balise de fin](#) a été omise. (Il ne peut pas être omis si l'élément est vide.)[colgroupcolcolgroup](#)

La [balise de fin](#)[colgroup](#) d'un élément peut être omise si l'élément n'est pas immédiatement suivi d' [un espace ASCII](#) ou d'un [commentaire](#) .[colgroup](#)

La [balise de fin](#)[caption](#) d'un élément peut être omise si l'élément n'est pas immédiatement suivi d' [un espace ASCII](#) ou d'un [commentaire](#) .[caption](#)

La [balise de fin](#)[thead](#) d'un élément peut être omise si l'élément est immédiatement suivi d'un élément ou [theadtbodytfoot](#)

La [balise de début](#)[tbody](#) d'un élément peut être omise si la première chose à l'intérieur de l'élément est un élément, et si l'élément n'est pas immédiatement précédé d'un élément , ou dont [la balise de fin](#) a été omise. (Il ne peut pas être omis si l'élément est vide.)[tbodytrtbodytheadtfoot](#)

La [balise de fin](#)[tbody](#) d'un élément peut être omise si l'élément est immédiatement suivi d'un élément ou , ou s'il n'y a plus de contenu dans l'élément parent.[tbodytbodytfoot](#)

La [balise de fin](#)[tfoot](#) d'un élément peut être omise s'il n'y a plus de contenu dans l'élément parent.

La [balise de fin](#)[tr](#) d'un élément peut être omise si l'élément est immédiatement suivi d'un autre élément, ou s'il n'y a plus de contenu dans l'élément parent.[trtr](#)

La [balise de fin](#)<sup>td</sup> d'un élément peut être omise si l'élément est immédiatement suivi d'un élément ou , ou s'il n'y a plus de contenu dans l'élément parent.[tdtdth](#)

La [balise de fin](#)<sup>th</sup> d'un élément peut être omise si l'élément est immédiatement suivi d'un élément ou , ou s'il n'y a plus de contenu dans l'élément parent.[thtdth](#)

La possibilité d'omettre toutes ces balises liées au tableau rend le balisage du tableau beaucoup plus concis.

Prenons cet exemple :

```
<table>
  <caption>37547 TEE Electric Powered Rail Car Train Functions
  (Abbreviated)</caption>
  <colgroup><col><col><col></colgroup>
  <thead>
    <tr>
      <th>Function</th>
      <th>Control Unit</th>
      <th>Central Station</th>
    </tr>
  </thead>
  <tbody>
    <tr>
      <td>Headlights</td>
      <td>✓</td>
      <td>✓</td>
    </tr>
    <tr>
      <td>Interior Lights</td>
      <td>✓</td>
      <td>✓</td>
    </tr>
    <tr>
      <td>Electric locomotive operating sounds</td>
      <td>✓</td>
      <td>✓</td>
    </tr>
    <tr>
      <td>Engineer's cab lighting</td>
      <td></td>
```

```
 ✓ || Station Announcements - Swiss |  | ✓ |

```

Le même tableau, modulo certaines différences d'espaces, pourrait être balisé comme suit :

```





```

```
 Station Announcements - Swiss |  | ✓ |
```

Étant donné que les cellules prennent beaucoup moins de place de cette façon, cela peut être encore plus concis en plaçant chaque ligne sur une ligne :

```

<table>
  <caption>37547 TEE Electric Powered Rail Car Train Functions
  (Abbreviated)
  <colgroup><col><col><col>
  <thead>
    <tr> <th>Function
    <th>Central Station
    <th>Control Unit
  </tr>
  <tbody>
    <tr> <td>Headlights
    <td>✓
    <td>✓
  </tr>
    <tr> <td>Interior Lights
    <td>✓
    <td>✓
  </tr>
    <tr> <td>Electric locomotive operating sounds
    <td>✓
    <td>✓
  </tr>
    <tr> <td>Engineer's cab lighting
    <td>
    <td>✓
  </tr>
    <tr> <td>Station Announcements - Swiss
    <td>
    <td>✓
  </tr>
</table>

```

Les seules différences entre ces tables, au niveau du DOM, résident dans la position précise de l'espace blanc (en tout cas sémantiquement neutre).

**Cependant** , une [balise de début](#) ne doit jamais être omise si elle possède des attributs.

Revenons à l'exemple précédent avec tous les espaces supprimés, puis toutes les balises facultatives supprimées :

```

<!DOCTYPE HTML><title>Hello</title><p>Welcome to this example.

```

Si l' [body](#) élément dans cet exemple devait avoir un [class](#) attribut et que l' [html](#) élément devait avoir un [lang](#) attribut, le balisage devrait devenir :

```

<!DOCTYPE HTML><html lang="en"><title>Hello</title><body
class="demo"><p>Welcome to this example.

```

*Cette section suppose que le document est conforme, en particulier qu'il n'y a pas de violation [du modèle de contenu](#) . L'omission de balises de la manière décrite dans*



*cette section dans un document qui n'est pas conforme aux [modèles de contenu](#) décrits dans cette spécification est susceptible d'entraîner des différences DOM inattendues (c'est en partie ce que les modèles de contenu sont conçus pour éviter).*

#### 13.1.2.5 Restrictions sur les modèles de contenu

Pour des raisons historiques, certains éléments ont des restrictions supplémentaires au-delà même des restrictions données par leur modèle de contenu.

Un [table](#) élément ne doit pas contenir [tr](#) d'éléments, même si ces éléments sont techniquement autorisés à l'intérieur [table](#) des éléments selon les modèles de contenu décrits dans cette spécification. (Si un [tr](#) élément est placé à l'intérieur de a [table](#) dans le balisage, cela impliquera en fait une [tbody](#) balise de début avant.)

Une seule [nouvelle ligne](#) peut être placée immédiatement après la [balise de début](#) des éléments [pre](#) et [textarea](#). Cela n'affecte pas le traitement de l'élément. Le [retour à la ligne autrement facultatif](#) doit être inclus si le contenu de l'élément lui-même commence par un [retour à la ligne](#) (car sinon le retour à la ligne de début dans le contenu serait traité comme le retour à la ligne facultatif et ignoré).

Les deux [pre](#) blocs suivants sont équivalents :

```
<pre>Hello</pre>
<pre>
Hello</pre>
```

#### 13.1.2.6 Restrictions sur le contenu du texte brut et des éléments de texte brut échappables

Le texte dans les éléments [de texte brut](#) et [les éléments de texte brut pouvant être échappés](#) ne doivent contenir aucune occurrence de la chaîne " " </ (U+003C SIGNE INFÉRIEUR À, U+002F SOLIDE) suivie de caractères qui correspondent sans distinction de casse au nom de balise de l'élément suivi de l'un des U+0009 TABULATION DE CARACTÈRES (tabulation), U+000A SAUT DE LIGNE (LF), U+000C SAUT DE FORME (FF), U+000D RETOUR CHARIOT (CR), U+0020 ESPACE, U+003E SIGNE SUPÉRIEUR À (>) ou U+002F SOLIDUS (/).

### 13.1.3 Texte

**Le texte** est autorisé à l'intérieur des éléments, des valeurs d'attribut et des commentaires. Des contraintes supplémentaires sont placées sur ce qui est et ce qui n'est pas autorisé dans le texte en fonction de l'endroit où le texte doit être placé, comme décrit dans les autres sections.

### 13.1.3.1 Nouvelles lignes

**Les retours à la ligne** en HTML peuvent être représentés soit par des caractères U+000D CARRIAGE RETURN (CR), des caractères U+000A LINE FEED (LF) ou des paires de caractères U+000D CARRIAGE RETURN (CR), U+000A LINE FEED (LF) dans cet ordre.

Là où [les références de caractère](#) sont autorisées, une référence de caractère d'un caractère U+000A LINE FEED (LF) (mais pas un caractère U+000D CARRIAGE RETURN (CR)) représente également une nouvelle [ligne](#) .

### 13.1.4 Références de caractères

Dans certains cas décrits dans d'autres sections, [le texte](#) peut être mélangé avec **des références de caractères** . Ceux-ci peuvent être utilisés pour échapper des caractères qui ne pourraient autrement pas être légalement inclus dans [text](#) .

Les références de caractères doivent commencer par un caractère U+0026 AMPERSAND (&). Ensuite, il existe trois types de références de caractères possibles :

#### Références de caractères nommés

L'esperluette doit être suivie de l'un des noms donnés dans la section [des références de caractères nommés](#) , en utilisant la même casse. Le nom doit se terminer par un caractère U+003B SEMICOLON (;).

#### Référence de caractère numérique décimal

L'esperluette doit être suivie d'un caractère NUMÉRO U+0023 (#), suivi d'un ou plusieurs [chiffres ASCII](#) , représentant un entier de base dix correspondant à un point de code autorisé selon la définition ci-dessous. Les chiffres doivent alors être suivis d'un caractère U+003B SEMICOLON (;).

#### Référence de caractère numérique hexadécimal

L'esperluette doit être suivie d'un caractère NUMÉRO U+0023 (#), qui doit être suivi soit d'un caractère U+0078 LETTRE MINUSCULE LATINE X (x) ou d'un caractère U+0058 LETTRE MAJUSCULE LATINE X (X), qui doit alors être suivi d'un ou plusieurs [chiffres hexadécimaux ASCII](#) , représentant un entier hexadécimal qui correspond à un point de code autorisé selon la

définition ci-dessous. Les chiffres doivent alors être suivis d'un caractère U+003B SEMICOLON (;).

Les formes de référence de caractères numériques décrites ci-dessus sont autorisées à référencer n'importe quel point de code à l'exception de U+000D CR, [des non-caractères](#) et [des contrôles](#) autres que [les espaces blancs ASCII](#) .

Une **esperluette ambiguë** est un caractère U+0026 AMPERSAND (&) suivi d'un ou plusieurs [caractères alphanumériques ASCII](#) , suivis d'un caractère U+003B SEMICOLON (;), où ces caractères ne correspondent à aucun des noms donnés dans le [caractère nommé rubrique références](#) .

### 13.1.5 Rubriques CDATA

**Les sections CDATA** doivent comprendre les composants suivants, dans cet ordre :

1. La chaîne " <![CDATA[".
2. En option, [text](#) , avec la restriction supplémentaire que le texte ne doit pas contenir la chaîne " ] ]>".
3. La chaîne " ] ]>".

Les sections CDATA ne peuvent être utilisées que dans du contenu étranger (MathML ou SVG). Dans cet exemple, une section CDATA est utilisée pour échapper le contenu d'un élément [MathML :ms](#)

```
<p>You can add a string to a number, but this stringifies the
number:</p>
<math>
  <ms><![CDATA[x<y]]></ms>
  <mo>+</mo>
  <mn>3</mn>
  <mo>=</mo>
  <ms><![CDATA[x<y3]]></ms>
</math>
```

### 13.1.6 Commentaires

**Les commentaires** doivent avoir le format suivant :

1. La chaîne " <!--".

2. Facultativement, [text](#), avec la restriction supplémentaire que le texte ne doit pas commencer par la chaîne " ">, ni commencer par la chaîne " ->", ni contenir les chaînes " <!--", " -->" ou " --!>", ni se terminer par la chaîne " <!--".
3. La chaîne " -->".

*Le [texte](#) peut se terminer par la chaîne " <!", comme dans <!--My favorite operators are > and <!-->.*

1.

## 1. [13.2 Analyser des documents HTML](#)

1. [13.2.1 Présentation du modèle d'analyse syntaxique](#)
2. [13.2.2 Erreurs d'analyse](#)
3. [13.2.3 Le flux d'octets d'entrée](#)
  1. [13.2.3.1 Analyse avec un codage de caractères connu](#)
  2. [13.2.3.2 Détermination du codage des caractères](#)
  3. [13.2.3.3 Encodages de caractères](#)
  4. [13.2.3.4 Modification de l'encodage lors de l'analyse](#)
  5. [13.2.3.5 Prétraitement du flux d'entrée](#)
4. [13.2.4 État d'analyse](#)
  1. [13.2.4.1 Le mode d'insertion](#)
  2. [13.2.4.2 L'empilement des éléments ouverts](#)
  3. [13.2.4.3 La liste des éléments de formatage actifs](#)
  4. [13.2.4.4 Les pointeurs d'éléments](#)
  5. [13.2.4.5 Autres drapeaux d'état d'analyse](#)
5. [13.2.5 Tokénisation](#)
  1. [13.2.5.1 État des données](#)
  2. [13.2.5.2 État RCDATA](#)
  3. [13.2.5.3 État TEXTE BRUT](#)
  4. [13.2.5.4 État des données de script](#)
  5. [13.2.5.5 État TEXTE CLAIR](#)
  6. [13.2.5.6 État ouvert de l'étiquette](#)
  7. [13.2.5.7 État ouvert d'étiquette de fin](#)
  8. [13.2.5.8 État du nom de variable](#)
  9. [13.2.5.9 État de signe inférieur à RCDATA](#)
  10. [13.2.5.10 État ouvert de l'étiquette de fin RCDATA](#)
  11. [13.2.5.11 État du nom de balise de fin RCDATA](#)
  12. [13.2.5.12 RAWTEXT état de signe inférieur à](#)
  13. [13.2.5.13 État ouvert de la balise de fin RAWTEXT](#)
  14. [13.2.5.14 État du nom de la balise de fin RAWTEXT](#)

15. [13.2.5.15 Etat des données de script inférieur à signe](#)
16. [13.2.5.16 État ouvert d'étiquette de fin de données de script](#)
17. [13.2.5.17 État du nom de balise de fin de données de script](#)
18. [13.2.5.18 Etat de début d'échappement des données de script](#)
19. [13.2.5.19 État du tiret de début d'échappement des données de script](#)
20. [13.2.5.20 État d'échappement des données de script](#)
21. [13.2.5.21 État du tiret échappé des données de script](#)
22. [13.2.5.22 Données de script échappées tiret état tiret](#)
23. [13.2.5.23 Les données de script ont échappé à l'état inférieur à signe](#)
24. [13.2.5.24 Etat ouvert de balise de fin échappée de données de script](#)
25. [13.2.5.25 Etat du nom de la balise de fin échappée des données de script](#)
26. [13.2.5.26 Etat de début d'échappement double des données de script](#)
27. [13.2.5.27 État d'échappement double des données de script](#)
28. [13.2.5.28 État du tiret à double échappement des données de script](#)
29. [13.2.5.29 Données de script tiret à double échappement état tiret](#)
30. [13.2.5.30 Les données de script ont échappé à deux fois à l'état de signe inférieur à](#)
31. [13.2.5.31 État final d'échappement double des données de script](#)
32. [13.2.5.32 Avant l'état du nom d'attribut](#)
33. [13.2.5.33 État du nom d'attribut](#)
34. [13.2.5.34 Après l'état du nom d'attribut](#)
35. [13.2.5.35 Avant l'état de la valeur d'attribut](#)
36. [13.2.5.36 État de la valeur d'attribut \(entre guillemets\)](#)
37. [13.2.5.37 État de la valeur d'attribut \(entre guillemets simples\)](#)
38. [13.2.5.38 État de valeur d'attribut \(sans guillemets\)](#)
39. [13.2.5.39 Après l'état de valeur d'attribut \(entre guillemets\)](#)
40. [13.2.5.40 État de la balise de début à fermeture automatique](#)
41. [13.2.5.41 État de commentaire erroné](#)
42. [13.2.5.42 État ouvert de la déclaration de balisage](#)
43. [13.2.5.43 Etat de début de commentaire](#)

- 44. [13.2.5.44 État du tiret de début de commentaire](#)
- 45. [13.2.5.45 État des commentaires](#)
- 46. [13.2.5.46 Commentaire état inférieur à signe](#)
- 47. [13.2.5.47 Commentaire état de bang signe inférieur à](#)
- 48. [13.2.5.48 Commentaire inférieur à signe bang état tiret](#)
- 49. [13.2.5.49 Commentaire signe inférieur à bang tiret tiret état](#)
- 50. [13.2.5.50 État du tiret de fin de commentaire](#)
- 51. [13.2.5.51 État final du commentaire](#)
- 52. [13.2.5.52 État du coup de fin de commentaire](#)
- 53. [13.2.5.53 État DOCTYPE](#)
- 54. [13.2.5.54 Avant l'état du nom DOCTYPE](#)
- 55. [13.2.5.55 État du nom de DOCTYPE](#)
- 56. [13.2.5.56 Après l'état du nom DOCTYPE](#)
- 57. [13.2.5.57 Après l'état du mot clé public DOCTYPE](#)
- 58. [13.2.5.58 Avant l'état de l'identificateur public DOCTYPE](#)
- 59. [13.2.5.59 État de l'identificateur public DOCTYPE \(entre guillemets\)](#)
- 60. [13.2.5.60 État de l'identificateur public DOCTYPE \(entre guillemets simples\)](#)
- 61. [13.2.5.61 Après l'état de l'identificateur public DOCTYPE](#)
- 62. [13.2.5.62 Entre l'état des identifiants public et système de DOCTYPE](#)
- 63. [13.2.5.63 Après l'état du mot clé système DOCTYPE](#)
- 64. [13.2.5.64 Avant l'état de l'identificateur de système DOCTYPE](#)
- 65. [13.2.5.65 État de l'identificateur de système DOCTYPE \(entre guillemets\)](#)
- 66. [13.2.5.66 État de l'identificateur système DOCTYPE \(entre guillemets simples\)](#)
- 67. [13.2.5.67 Après l'état de l'identificateur de système DOCTYPE](#)
- 68. [13.2.5.68 État DOCTYPE erroné](#)
- 69. [13.2.5.69 État de la section CDATA](#)
- 70. [13.2.5.70 État du support de section CDATA](#)
- 71. [13.2.5.71 État final de la section CDATA](#)
- 72. [13.2.5.72 Etat de référence des caractères](#)
- 73. [13.2.5.73 Etat de référence du caractère nommé](#)
- 74. [13.2.5.74 Esperluette ambiguë](#)
- 75. [13.2.5.75 Etat de référence des caractères numériques](#)
- 76. [13.2.5.76 Etat de début de référence de caractère hexadécimal](#)
- 77. [13.2.5.77 Etat de début de référence de caractère décimal](#)

- 78. [13.2.5.78 Etat de référence des caractères hexadécimaux](#)
- 79. [13.2.5.79 Etat de référence du caractère décimal](#)
- 80. [13.2.5.80 Etat final de référence de caractère numérique](#)
- 6. [13.2.6 Construction d'arbres](#)
  - 1. [13.2.6.1 Création et insertion de nœuds](#)
  - 2. [13.2.6.2 Analyser des éléments qui ne contiennent que du texte](#)
  - 3. [13.2.6.3 Éléments de fermeture qui ont des balises de fin implicites](#)
  - 4. [13.2.6.4 Les règles d'analyse des jetons dans le contenu HTML](#)
    - 1. [13.2.6.4.1 Le mode d'insertion "initial"](#)
    - 2. [13.2.6.4.2 Le mode d'insertion "avant html"](#)
    - 3. [13.2.6.4.3 Le mode d'insertion "avant tête"](#)
    - 4. [13.2.6.4.4 Le mode d'insertion "en tête"](#)
    - 5. [13.2.6.4.5 Le mode d'insertion "in head noscript"](#)
    - 6. [13.2.6.4.6 Le mode d'insertion "après tête"](#)
    - 7. [13.2.6.4.7 Le mode d'insertion "dans le corps"](#)
    - 8. [13.2.6.4.8 Le mode d'insertion "texte"](#)
    - 9. [13.2.6.4.9 Le mode d'insertion "dans le tableau"](#)
    - 10. [13.2.6.4.10 Le mode d'insertion "dans le texte du tableau"](#)
    - 11. [13.2.6.4.11 Le mode d'insertion "en légende"](#)
    - 12. [13.2.6.4.12 Le mode d'insertion "dans le groupe de colonnes"](#)
    - 13. [13.2.6.4.13 Le mode d'insertion "dans le corps du tableau"](#)
    - 14. [13.2.6.4.14 Le mode d'insertion "en ligne"](#)
    - 15. [13.2.6.4.15 Le mode d'insertion "dans cellule"](#)
    - 16. [13.2.6.4.16 Le mode d'insertion "en sélection"](#)
    - 17. [13.2.6.4.17 Le mode d'insertion "dans select in table"](#)
    - 18. [13.2.6.4.18 Le mode d'insertion "dans le modèle"](#)
    - 19. [13.2.6.4.19 Le mode d'insertion "après le corps"](#)
    - 20. [13.2.6.4.20 Le mode d'insertion "in frameset"](#)
    - 21. [13.2.6.4.21 Le mode d'insertion "après frameset"](#)
    - 22. [13.2.6.4.22 Le mode d'insertion "après après corps"](#)
    - 23. [13.2.6.4.23 Le mode d'insertion "après après frameset"](#)
  - 5. [13.2.6.5 Les règles d'analyse des jetons dans le contenu étranger](#)
- 7. [13.2.7 La fin](#)
- 8. [13.2.8 Analyse HTML spéculative](#)
- 9. [13.2.9 Contrainte d'un DOM HTML dans un ensemble d'informations](#)



10. [13.2.10 Une introduction à la gestion des erreurs et des cas étranges dans l'analyseur](#)
  1. [13.2.10.1 Balises mal imbriquées : `<b><i></b></i>`](#)
  2. [13.2.10.2 Balises mal imbriquées : `<b><p></b></p>`](#)
  3. [13.2.10.3 Balisage inattendu dans les tableaux](#)
  4. [13.2.10.4 Scripts qui modifient la page lors de son analyse](#)
  5. [13.2.10.5 L'exécution de scripts qui se déplacent sur plusieurs documents](#)
  6. [13.2.10.6 Éléments de formatage non fermés](#)
2. [13.3 Sérialisation de fragments HTML](#)
3. [13.4 Analyser des fragments HTML](#)

## 13.2 Analyser des documents HTML

*Cette section s'applique uniquement aux agents utilisateurs, aux outils d'exploration de données et aux vérificateurs de conformité.*

*Les règles d'analyse des documents XML dans les arbres DOM sont couvertes par la section suivante, intitulée " [La syntaxe XML](#) ".*

Les agents utilisateurs doivent utiliser les règles d'analyse décrites dans cette section pour générer les arbres DOM à partir [text/html](#) des ressources. Ensemble, ces règles définissent ce que l'on appelle l' **analyseur HTML** .

*Bien que la syntaxe HTML décrite dans cette spécification ressemble beaucoup à SGML et XML, il s'agit d'un langage distinct avec ses propres règles d'analyse.*

*Certaines versions antérieures de HTML (en particulier de HTML2 à HTML4) étaient basées sur SGML et utilisaient des règles d'analyse SGML. Cependant, peu de navigateurs Web (le cas échéant) ont implémenté une véritable analyse SGML pour les documents HTML ; les seuls agents utilisateurs à gérer strictement HTML comme une application SGML ont été historiquement des validateurs. La confusion qui en résulte - avec des validateurs affirmant que les documents ont une représentation alors que des navigateurs Web largement déployés implémentent de manière interopérable une représentation différente - a gaspillé des décennies de productivité. Cette version de HTML revient donc à une base non SGML.*

*Les auteurs intéressés par l'utilisation des outils SGML dans leur pipeline de création sont encouragés à utiliser les outils XML et la sérialisation XML du HTML.*

Pour les besoins des vérificateurs de conformité, si une ressource est déterminée comme étant dans [la syntaxe HTML](#) , il s'agit alors d'un [document HTML](#) .



Comme indiqué [dans la section de terminologie](#) , les références aux [types d'éléments](#) qui ne spécifient pas explicitement un espace de noms font toujours référence aux éléments de l' [espace de noms HTML](#) . Par exemple, si la spécification parle d'"un [menu](#) élément", alors c'est un élément avec le nom local "[menu](#)", l'espace de noms "<http://www.w3.org/1999/xhtml>" et l'interface [HTMLMenuElement](#). Dans la mesure du possible, les références à ces éléments sont hyperliées à leur définition.

### 13.2.1 Présentation du modèle d'analyse syntaxique

L'entrée du processus d'analyse HTML consiste en un flux de [points de code](#) , qui passe par une étape [de tokenisation](#) suivie d'une étape [de construction d'arborescence](#) . La sortie est un [Document](#) objet.

Les implémentations qui [ne prennent pas en charge les scripts](#) n'ont pas besoin de créer un [Document](#) objet DOM, mais l'arborescence DOM dans de tels cas est toujours utilisée comme modèle pour le reste de la spécification.

Dans le cas courant, les données gérées par l'étape de tokenisation proviennent du réseau, mais [elles peuvent également provenir d'un script](#) exécuté dans l'agent utilisateur, par exemple à l'aide de l' `document.write()` API.

Il n'y a qu'un seul ensemble d'états pour l'étape du tokenizer et l'étape de construction de l'arbre, mais l'étape de construction de l'arbre est réentrante, ce qui signifie que pendant que l'étape de construction de l'arbre gère un jeton, le tokenizer peut être repris, provoquant l'émission d'autres jetons et traité avant que le traitement du premier jeton ne soit terminé.

Dans l'exemple suivant, l'étape de construction de l'arborescence sera appelée à gérer un jeton de balise de début "p" tout en gérant le jeton de balise de fin "script" :

```
...  
<script>  
  document.write('<p>');  
</script>  
...
```

Pour gérer ces cas, les analyseurs ont un **niveau d'imbrication de script** , qui doit être initialement défini sur zéro, et un **indicateur de pause d'analyseur** , qui doit être initialement défini sur faux.

### 13.2.2 Erreurs d'analyse

Cette spécification définit les règles d'analyse des documents HTML, qu'ils soient syntaxiquement corrects ou non. Certains points de l'algorithme d'analyse sont dits être [des erreurs d'analyse](#) . La gestion des erreurs d'analyse est bien définie (ce sont les règles de traitement décrites tout au long de cette spécification), mais les agents utilisateurs, lors de l'analyse d'un document HTML, peuvent interrompre l'analyse à la première erreur d'analyse [qu'ils](#) rencontrent [et](#) pour laquelle ils ne souhaitent pas appliquer les règles décrites dans cette spécification.

Les vérificateurs de conformité doivent signaler au moins une condition d'erreur d'analyse à l'utilisateur si une ou plusieurs conditions d'erreur d'analyse existent dans le document et ne doivent pas signaler les conditions d'erreur d'analyse s'il n'en existe aucune dans le document. Les vérificateurs de conformité peuvent signaler plusieurs conditions d'erreur d'analyse si plusieurs conditions d'erreur d'analyse existent dans le document.

*Les erreurs d'analyse ne sont que des erreurs avec la syntaxe du HTML. En plus de vérifier les erreurs d'analyse, les vérificateurs de conformité vérifieront également que le document obéit à toutes les autres exigences de conformité décrites dans la présente spécification.*

Certaines erreurs d'analyse ont des codes dédiés décrits dans le tableau ci-dessous qui doivent être utilisés par les vérificateurs de conformité dans les rapports.

*Les descriptions d'erreurs dans le tableau ci-dessous ne sont pas normatives.*

Code	Description
<b>fermeture-brutale-de-commentaire- vide</b>	Cette erreur se produit si l'analyseur rencontre un <a href="#">commentaire</a> vide qui est brusquement fermé par un <a href="#">point de code</a> U+003E (>) (c'est-à-dire <!-->ou <!-->). L'analyseur se comporte comme si le commentaire était correctement fermé.
<b>abrupt-doctype- public-identifier</b>	<a href="#">Cette erreur se produit si l'analyseur rencontre un point de code</a> U+003E (>) dans l'identificateur public <a href="#">DOCTYPE</a> <!DOCTYPE html PUBLIC "foo"> (par exemple, ). Dans un tel cas, si le DOCTYPE est correctement placé en tant que préambule de document, l'analyseur définit le <a href="#">modeDocument</a> bizarrerie .
<b>identificateur de système de type de document abrupt</b>	<a href="#">Cette erreur se produit si l'analyseur rencontre un point de code</a> U+003E (>) dans l'identifiant système <a href="#">DOCTYPE</a> (par exemple, <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN" "foo">). Dans un tel cas, si le DOCTYPE est correctement placé en tant que préambule de document, l'analyseur définit le <a href="#">modeDocument</a> bizarrerie .
<b>absence-de- chiffres-dans-la- référence-de- caractère- numérique</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">référence de caractère</a> numérique qui ne contient aucun chiffre (par exemple, &#qux;). Dans ce cas, l'analyseur ne résout pas la référence de caractère.
<b>cdata-dans-le- contenu-html</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">section CDATA</a> en dehors du contenu étranger (SVG ou MathML). L'analyseur traite ces sections CDATA (y compris les chaînes de début " [CDATA[" et de fin " ] ]") comme des commentaires.
<b>référence-de- caractère-en- dehors-de-la- plage-unicode</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">référence de caractère</a> numérique qui fait référence à un <a href="#">point de code</a> supérieur à la plage Unicode valide. L'analyseur résout une telle référence de caractère en un CARACTERE DE REMPLACEMENT U+FFFD.
<b>caractère-de- contrôle-dans-le- flux-d'entrée</b>	Cette erreur se produit si le <a href="#">flux d'entrée</a> contient un <a href="#">point de code de contrôle</a> qui n'est pas <a href="#">un espace blanc ASCII</a> ou U+0000 NULL. De tels points de code sont analysés tels

Code	Description
	quels et généralement, lorsque les règles d'analyse n'appliquent aucune restriction supplémentaire, se retrouvent dans le DOM.
<b>référence du caractère de contrôle</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">référence de caractère</a> numérique qui fait référence à un <a href="#">point de code de contrôle</a> qui n'est pas <a href="#">un espace blanc ASCII</a> ou qui est un retour chariot U+000D. L'analyseur résout ces références de caractère telles quelles, à l'exception des références de contrôle C1 qui sont remplacées en fonction de l' <a href="#">état final de la référence de caractère numérique</a> .
<b>balise de fin avec attributs</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">balise de fin</a> avec <a href="#">des attributs</a> . Les attributs dans les balises de fin sont ignorés et n'entrent pas dans le DOM.
<b>attribut dupliqué</b>	Cette erreur se produit si l'analyseur rencontre un <a href="#">attribut</a> dans une balise qui a déjà un attribut avec le même nom. L'analyseur ignore toutes ces occurrences en double de l'attribut.
<b>balise-de-fin-avec-fin-solidus</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">balise de fin qui a un point de code</a> U+002F (/) juste avant le point de code de fermeture U+003E (>) (par exemple, </div/>). Une telle balise est traitée comme une balise de fin normale.
<b>eof-before-tag-name</b>	Cette erreur se produit si l'analyseur rencontre la fin du <a href="#">flux d'entrée</a> où un nom de balise est attendu. Dans ce cas, l'analyseur traite le début d'une <a href="#">balise de début</a> (c'est-à-dire <) ou une <a href="#">balise de fin</a> (c'est-à-dire </) comme du contenu textuel.
<b>eof-in-cdata</b>	Cette erreur se produit si l'analyseur rencontre la fin du <a href="#">flux d'entrée</a> dans une <a href="#">section CDATA</a> . L'analyseur traite ces sections CDATA comme si elles étaient fermées immédiatement avant la fin du flux d'entrée.
<b>eof-en-commentaire</b>	Cette erreur se produit si l'analyseur rencontre la fin du <a href="#">flux d'entrée</a> dans un <a href="#">commentaire</a> . L'analyseur traite ces commentaires comme s'ils étaient fermés immédiatement avant la fin du flux d'entrée.
<b>eof-dans-doctype</b>	Cette erreur se produit si l'analyseur rencontre la fin du flux d'entrée dans un <a href="#">DOCTYPE</a> . Dans un tel cas, si le

Code	Description
	DOCTYPE est correctement placé en tant que préambule de document, l'analyseur définit le <a href="#">modeDocument</a> bizarrerie .
eof-in-script-html-comment-like-text	<p>Cette erreur se produit si l'analyseur rencontre la fin du <a href="#">flux d'entrée</a> dans un texte qui ressemble à un <a href="#">commentaire HTML</a> dans <a href="#">script</a>le contenu de l'élément (par exemple, <code>&lt;script&gt;&lt;!-- foo</code>).</p> <p><i>Les structures syntaxiques qui ressemblent à des commentaires HTML dans <a href="#">script</a> les éléments sont analysées comme du contenu textuel. Ils peuvent faire partie d'une structure syntaxique spécifique au langage de script ou être traités comme un commentaire de type HTML, si le langage de script les prend en charge (par exemple, les règles d'analyse pour les commentaires de type HTML peuvent être trouvées dans l'annexe B de la spécification JavaScript) . La raison courante de cette erreur est une violation des <a href="#">restrictions sur le contenu des script éléments</a> . <a href="#">[JAVASCRIPT]</a></i></p>
eof-in-tag	Cette erreur se produit si l'analyseur rencontre la fin du <a href="#">flux d'entrée</a> dans une <a href="#">balise de début</a> ou une <a href="#">balise de fin</a> (par exemple, <code>&lt;div id=</code> ). Une telle balise est ignorée.
commentaire mal fermé	Cette erreur se produit si l'analyseur rencontre un <a href="#">commentaire</a> fermé par la séquence <a href="#">de points de code</a> <code>--!&gt; " "</code> . L'analyseur traite ces commentaires comme s'ils étaient correctement fermés par la séquence de points de code <code>" ".--&gt;</code>
commentaire mal ouvert	<p>Cette erreur se produit si l'analyseur rencontre la <code>" &lt;!"</code> séquence <a href="#">de points de code</a> qui n'est pas immédiatement suivie de deux points de code U+002D (-) et qui n'est pas le début d'une <a href="#">section DOCTYPE</a> ou CDATA . Tout le contenu qui suit la séquence de points de code <code>" "</code> jusqu'à un point de code U+003E (&gt;) (le cas échéant) ou jusqu'à la fin du <a href="#">flux d'entrée</a> est traité comme un commentaire.&lt;!</p> <p><i>Une cause possible de cette erreur est l'utilisation d'une déclaration de balisage XML (par exemple, <code>&lt;!ELEMENT br EMPTY&gt;</code>) en HTML.</i></p>
séquence-de-caractères-invalide-après-le-nom-de-doctype	Cette erreur se produit si l'analyseur rencontre une séquence <a href="#">de points de code</a> autre que les mots-clés <code>" PUBLIC"</code> et <code>" SYSTEM"</code> après un nom <a href="#">DOCTYPE</a> . Dans un tel cas, l'analyseur ignore tous les identifiants publics ou système suivants, et si le DOCTYPE est correctement placé en tant que préambule de document, et si l' <a href="#">analyseur ne peut pas</a>

Code	Description
	<a href="#">changer l'indicateur de mode</a> est faux, définit le <a href="#">modeDocument</a> bizarrerie .
invalid-first-character-of-tag-name	<p>Cette erreur se produit si l'analyseur rencontre un <a href="#">point de code</a> qui n'est pas un <a href="#">alpha ASCII</a> où le premier point de code d'un nom <a href="#">de balise de début</a> ou un nom <a href="#">de balise de fin</a> est attendu. Si une balise de début était attendue, un tel point de code et un U+003C précédent (&lt;) sont traités comme du contenu textuel, et tout le contenu qui suit est traité comme du balisage. Alors que, si une balise de fin était attendue, ce point de code et tout le contenu qui suit jusqu'à un point de code U+003E (&gt;) (le cas échéant) ou jusqu'à la fin du flux d'entrée <a href="#">est</a> traité comme un commentaire.</p> <p>Par exemple, considérez le balisage suivant :</p> <pre>&lt;42&gt;&lt;/42&gt;</pre> <p>Cela sera analysé en :</p> <ul style="list-style-type: none"> <li><a href="#">html</a> <ul style="list-style-type: none"> <li><a href="#">head</a></li> <li><a href="#">body</a> <ul style="list-style-type: none"> <li><a href="#">#text</a>: &lt;42&gt;</li> <li><a href="#">#comment</a>: 42</li> </ul> </li> </ul> </li> </ul> <p><i>Alors que le premier point de code d'un nom de balise est limité à un <a href="#">ASCII alpha</a> , un large éventail de points de code (y compris <a href="#">les chiffres ASCII</a> ) est autorisé dans les positions suivantes.</i></p>
valeur d'attribut manquante	<a href="#">Cette erreur se produit si l'analyseur rencontre un point de code</a> U+003E (>) où une valeur <a href="#">d'attribut</a> <code>&lt;div id=&gt;</code> est attendue (par exemple, ). L'analyseur traite l'attribut comme ayant une valeur vide.
nom-de-type-de-doc-manquant	Cette erreur se produit si l'analyseur rencontre un <a href="#">DOCTYPE</a> auquel il manque un nom (par exemple, <code>&lt;!DOCTYPE&gt;</code> ). Dans un tel cas, si le DOCTYPE est correctement placé en tant que préambule de document, l'analyseur définit le <a href="#">modeDocument</a> bizarrerie .
manquant-doctype-public-identifiant	<a href="#">Cette erreur se produit si l'analyseur rencontre un point de code</a> U+003E (>) où le début de l' identificateur public <a href="#">DOCTYPE</a> <code>&lt;!DOCTYPE html PUBLIC &gt;</code> est attendu (par exemple, ). Dans un tel cas, si le DOCTYPE est correctement

Code	Description
	placé en tant que préambule de document, l'analyseur définit le <a href="#">modeDocument</a> bizarrerie .
identificateur-de-système-de-type-de-document-manquant	<a href="#">Cette erreur se produit si l'analyseur rencontre un point de code U+003E (&gt;) où le début de l' identifiant système DOCTYPE</a> <!DOCTYPE html SYSTEM > est attendu (par exemple, ). Dans un tel cas, si le DOCTYPE est correctement placé en tant que préambule de document, l'analyseur définit le <a href="#">modeDocument</a> bizarrerie .
nom de balise de fin manquant	<a href="#">Cette erreur se produit si l'analyseur rencontre un point de code U+003E (&gt;) où un nom de balise de fin</a> est attendu, c'est-à-dire </>. L'analyseur ignore toute la </>séquence de points de code " " .
missing-quote-before-doctype-public-identifier	Cette erreur se produit si l'analyseur rencontre l' identificateur public <a href="#">DOCTYPE</a> qui n'est pas précédé d'un guillemet (par exemple, <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01//EN">). Dans un tel cas, l'analyseur ignore l'identifiant public et, si le DOCTYPE est correctement placé en tant que préambule du document, définit le <a href="#">modeDocument</a> bizarrerie .
missing-quote-before-doctype-system-identifier	Cette erreur se produit si l'analyseur rencontre l' identificateur système <a href="#">DOCTYPE</a> qui n'est pas précédé d'un guillemet (par exemple, <!DOCTYPE html SYSTEM http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">). Dans un tel cas, l'analyseur ignore l'identifiant système et, si le DOCTYPE est correctement placé en tant que préambule du document, définit le <a href="#">modeDocument</a> bizarrerie .
point-virgule-manquant-après-référence-de-caractère	<p>Cette erreur se produit si l'analyseur rencontre une <a href="#">référence de caractère</a> qui ne se termine pas par un <a href="#">point de code U+003B (;)</a> . Habituellement, l'analyseur se comporte comme si la référence de caractère se terminait par le point de code U+003B (;) ; cependant, il existe des cas ambigus dans lesquels l'analyseur inclut des points de code suivants dans la référence de caractère.</p> <p>Par exemple, &amp;not;insera analysé comme " -in" alors que &amp;notinsera analysé comme " €".</p>
missing-whitespace-after-doctype-public-keyword	Cette erreur se produit si l' analyseur rencontre un <a href="#">DOCTYPE</a> dont " PUBLIC" le mot - clé et l' identifiant public ne sont pas séparés par <a href="#">des espaces ASCII</a> . Dans ce cas,



Code	Description
	l'analyseur se comporte comme si un espace blanc ASCII était présent.
<b>missing-whitespace-after-doctype-system-keyword</b>	Cette erreur se produit si l' analyseur rencontre un <a href="#">DOCTYPE</a> dont " <small>SYSTEM</small> " le mot - clé et l' identificateur système ne sont pas séparés par <a href="#">des espaces ASCII</a> . Dans ce cas, l'analyseur se comporte comme si un espace blanc ASCII était présent.
<b>missing-whitespace-before-doctype-name</b>	Cette erreur se produit si l'analyseur rencontre un <a href="#">DOCTYPE</a> dont le " <small>DOCTYPE</small> " mot-clé et le nom ne sont pas séparés par <a href="#">des espaces ASCII</a> . Dans ce cas, l'analyseur se comporte comme si un espace blanc ASCII était présent.
<b>espace-blanc-manquant-entre-attributs</b>	Cette erreur se produit si l'analyseur rencontre <a href="#">des attributs</a> qui ne sont pas séparés par <a href="#">des espaces blancs ASCII</a> (par exemple, <code>&lt;div id="foo"class="bar"&gt;</code> ). Dans ce cas, l'analyseur se comporte comme si un espace blanc ASCII était présent.
<b>espace-blanc-manquant-entre-doctype-public-et-identificateurs-système</b>	Cette erreur se produit si l'analyseur rencontre un <a href="#">DOCTYPE</a> dont les identifiants public et système ne sont pas séparés par <a href="#">des espaces ASCII</a> . Dans ce cas, l'analyseur se comporte comme si un espace blanc ASCII était présent.
<b>commentaire imbriqué</b>	Cette erreur se produit si l'analyseur rencontre un <a href="#">commentaire</a> imbriqué (par exemple, <code>&lt;!-- &lt;!-- nested --&gt; --&gt;</code> ). Un tel commentaire sera fermé par la première séquence <a href="#">de points de code</a> <code>--&gt;</code> " " et tout ce qui suit sera traité comme du balisage.
<b>référence-de-caractère-non-caractère</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">référence de caractère</a> numérique qui fait référence à un <a href="#">non-caractère</a> . L'analyseur résout ces références de caractères telles quelles.
<b>noncharacter-in-input-stream</b>	Cette erreur se produit si le <a href="#">flux d'entrée</a> contient un <a href="#">noncharacter</a> . De tels <a href="#">points de code</a> sont analysés tels quels et généralement, lorsque les règles d'analyse n'appliquent aucune restriction supplémentaire, se retrouvent dans le DOM.



Code	Description
<b>non-void-html-element-start-tag-with-trailing-solidus</b>	<p>Cette erreur se produit si l'analyseur rencontre une <a href="#">balise de début</a> pour un élément qui n'est pas dans la liste des <a href="#">éléments vides</a> ou qui ne fait pas partie du contenu étranger (c'est-à-dire, pas un élément SVG ou MathML) qui a un <a href="#">code U+002F (/) point</a> juste avant le point de code de fermeture U+003E (&gt;). L'analyseur se comporte comme si le U+002F (/) n'était pas présent.</p> <p>Par exemple, considérez le balisage suivant :</p> <pre>&lt;div/&gt;&lt;span&gt;&lt;/span&gt;&lt;span&gt;&lt;/span&gt;</pre> <p>Cela sera analysé en :</p> <ul style="list-style-type: none"> <li>• <a href="#">html</a> <ul style="list-style-type: none"> <li>○ <a href="#">head</a></li> <li>○ <a href="#">body</a> <ul style="list-style-type: none"> <li>▪ <a href="#">div</a> <ul style="list-style-type: none"> <li>▪ <a href="#">span</a></li> <li>▪ <a href="#">span</a></li> </ul> </li> </ul> </li> </ul> </li> </ul> <p><i>Le U+002F (/) à la fin d'un nom de balise de début ne peut être utilisé que dans un contenu étranger pour spécifier des balises à fermeture automatique. (Les balises à fermeture automatique n'existent pas en HTML.) Elles sont également autorisées pour les éléments vides, mais n'ont aucun effet dans ce cas.</i></p>
<b>null-character-reference</b>	<p>Cette erreur se produit si l'analyseur rencontre une <a href="#">référence de caractère numérique qui fait référence à un point de code</a> U+0000 NULL . L'analyseur résout ces références de caractères en un CARACTÈRE DE REMPLACEMENT U+FFFD.</p>
<b>référence-de-caractère-de-substitution</b>	<p>Cette erreur se produit si l'analyseur rencontre une <a href="#">référence de caractère</a> numérique qui fait référence à un <a href="#">substitut</a> . L'analyseur résout ces références de caractères en un CARACTÈRE DE REMPLACEMENT U+FFFD.</p>
<b>substitut dans le flux d'entrée</b>	<p>Cette erreur se produit si le <a href="#">flux d'entrée</a> contient un <a href="#">substitut</a> . De tels <a href="#">points de code</a> sont analysés tels quels et généralement, lorsque les règles d'analyse n'appliquent aucune restriction supplémentaire, se retrouvent dans le DOM.</p> <p><i>Les substituts ne peuvent trouver leur chemin dans le flux d'entrée que via des API de script telles que <code>document.write()</code>.</i></p>

Code	Description
caractère-inattendu-après-doctype-system-identifier	<p>Cette erreur se produit si l'analyseur rencontre des <a href="#">points de code</a> autres que <a href="#">les espaces blancs ASCII</a> ou la fermeture U+003E (&gt;) après l'identificateur système <a href="#">DOCTYPE</a>. L'analyseur ignore ces points de code.</p>
caractère-inattendu-dans-le-nom-de-l'attribut	<p><a href="#">Cette erreur se produit si l'analyseur rencontre un point de code</a> U+0022 ("), U+0027 (') ou U+003C (&lt;) dans un <a href="#">nom d'attribut</a>. L'analyseur inclut ces points de code dans le nom d'attribut.</p> <p><i>Les points de code qui déclenchent cette erreur font généralement partie d'une autre construction syntaxique et peuvent être le signe d'une faute de frappe autour du nom de l'attribut.</i></p> <p>Par exemple, considérez le balisage suivant :</p> <pre>&lt;div foo&lt;div&gt;</pre> <p>En raison d'un point de code U+003E (&gt;) oublié après <code>foo</code> que l'analyseur traite ce balisage comme un <code>div</code> élément unique avec un <code>foo&lt;div</code> attribut " ".</p> <p>Comme autre exemple de cette erreur, considérez le balisage suivant :</p> <pre>&lt;div id'bar'&gt;</pre> <p>En raison d'un point de code U+003D (=) oublié entre un nom d'attribut et une valeur, l'analyseur traite ce balisage comme un <code>div</code> élément avec l'attribut " <code>id'bar'</code> " qui a une valeur vide.</p>
caractère-inattendu-dans-la-valeur-d'attribut-sans-guillemets	<p>Cette erreur se produit si l'analyseur rencontre un point de code U+0022 ("), U+0027 ('), U+003C (&lt;), U+003D (=) ou U+0060 (`) dans une valeur d'attribut <a href="#">sans guillemets</a>. L'analyseur inclut ces points de code dans la valeur de l'attribut.</p> <p><i>Les points de code qui déclenchent cette erreur font généralement partie d'une autre construction syntaxique et peuvent être le signe d'une faute de frappe autour de la valeur de l'attribut.</i></p> <p><i>U+0060 (`) figure dans la liste des points de code qui déclenchent cette erreur car certains agents utilisateurs hérités la traitent comme une citation.</i></p> <p>Par exemple, considérez le balisage suivant :</p> <pre>&lt;div foo=b'ar'&gt;</pre>

Code	Description
	En raison d'un point de code U+0027 (') mal placé, l'analyseur définit la valeur de l'attribut " " sur "b'ar'".
<b>signe-égal-inattendu-devant-le-nom-de-l'attribut</b>	<p>Cette erreur se produit si l'analyseur rencontre un <a href="#">point de code</a> U+003D (=) avant un nom d'attribut. Dans ce cas, l'analyseur traite U+003D (=) comme le premier point de code du nom d'attribut.</p> <p><i>La raison courante de cette erreur est un nom d'attribut oublié.</i></p> <p>Par exemple, considérez le balisage suivant :</p> <pre>&lt;div foo="bar" ="baz"&gt;</pre> <p>En raison d'un nom d'attribut oublié, l'analyseur traite ce balisage comme un <a href="#">div</a> élément avec deux attributs : un attribut " " avec une valeur " " et un ="baz"attribut " " avec une valeur vide.</p>
<b>caractère nul inattendu</b>	Cette erreur se produit si l'analyseur rencontre un <a href="#">point de code</a> U+0000 NULL dans le <a href="#">flux d'entrée</a> à certaines positions. En général, ces points de code sont soit ignorés soit, pour des raisons de sécurité, remplacés par un CARACTÈRE DE REMPLACEMENT U+FFFD.
<b>point-d'interrogation-inattendu-au-lieu-du-nom-de-la-balise</b>	<p>Cette erreur se produit si l'analyseur rencontre un <a href="#">point de code</a> U+003F (?) où le premier point de code d'un nom <a href="#">de balise de début</a> est attendu. Le U+003F (?) et tout le contenu qui suit jusqu'à un point de code U+003E (&gt;) (le cas échéant) ou jusqu'à la fin du <a href="#">flux d'entrée</a> est traité comme un commentaire.</p> <p>Par exemple, considérez le balisage suivant :</p> <pre>&lt;?xml-stylesheet type="text/css" href="style.css"?)</pre> <p>Cela sera analysé en :</p> <ul style="list-style-type: none"> <li>• <a href="#">#comment</a>: ?xml-stylesheet type="text/css" href="style.css" ?</li> <li>• <a href="#">html</a> <ul style="list-style-type: none"> <li>◦ <a href="#">head</a></li> <li>◦ <a href="#">body</a></li> </ul> </li> </ul> <p><i>La raison courante de cette erreur est une instruction de traitement XML (par exemple, &lt;?xml-stylesheet type="text/css" href="style.css"?) ou une déclaration</i></p>

Code	Description
	<i>XML (par exemple, <code>&lt;?xml version="1.0" encoding="UTF-8" ?&gt;</code>) utilisée en HTML.</i>
<b>inattendu-solidus-dans-tag</b>	Cette erreur se produit si l'analyseur rencontre un <a href="#">point de code</a> U+002F (/) qui ne fait pas partie d'une valeur <a href="#">d'attribut</a> entre guillemets et n'est pas immédiatement suivi d'un point de code U+003E (>) dans une balise (par exemple, <code>&lt;div / id="foo"&gt;</code> ). Dans ce cas, l'analyseur se comporte comme s'il rencontrait <a href="#">des espaces blancs ASCII</a> .
<b>référence-de-personnage-nommée-inconnue</b>	Cette erreur se produit si l'analyseur rencontre une <a href="#">esperluette ambiguë</a> . Dans ce cas, l'analyseur ne résout pas la <a href="#">référence de caractère</a> .

### 13.2.3 Le flux d'octets d'entrée

Le flux de points de code qui comprend l'entrée de l'étape de tokenisation sera initialement vu par l'agent utilisateur comme un flux d'octets (provenant généralement du réseau ou du système de fichiers local). Les octets codent les caractères réels selon un *codage de caractères* particulier , que l'agent utilisateur utilise pour décoder les octets en caractères.

*Pour les documents XML, l'algorithme que les agents utilisateurs doivent utiliser pour déterminer le codage des caractères est donné par XML . Cette section ne s'applique pas aux documents XML. [\[XML\]](#)*

Habituellement, l' [algorithme de reniflage de codage](#) défini ci-dessous est utilisé pour déterminer le codage des caractères.

Étant donné un codage de caractères, les octets du [flux d'octets d'entrée](#) doivent être convertis en caractères pour le [flux d'entrée](#) du tokenizer , en transmettant le [flux d'octets d'entrée](#) et le codage de caractères à [decode](#) .

*Une marque d'ordre d'octet (BOM) de tête entraîne l'ignorance de l'argument de codage de caractères et sera lui-même ignoré.*

*Les octets ou séquences d'octets dans le flux d'octets d'origine qui n'étaient pas conformes à la norme de codage (par exemple, des séquences d'octets UTF-8 non valides dans un flux d'octets d'entrée UTF-8) sont des erreurs que les vérificateurs de conformité sont censés signaler. [\[CODAGE\]](#)*

***Les algorithmes du décodeur décrivent comment gérer les entrées non valides ; pour des raisons de sécurité, il est impératif que ces règles soient suivies à la lettre. Les différences dans la manière dont les séquences d'octets non valides sont gérées peuvent entraîner, entre autres problèmes, des vulnérabilités d'injection de script ("XSS").***

Lorsque l'analyseur HTML décode un flux d'octets d'entrée, il utilise un codage de caractères et un **confiance** . La confiance est soit *provisoire* , *certaine* ou *non pertinente* . L'encodage utilisé, et si la confiance dans cet encodage est *provisoire* ou *certaine* , est [utilisé pendant l'analyse](#) pour déterminer s'il faut [changer l'encodage](#) . Si aucun encodage n'est nécessaire, par exemple parce que l'analyseur fonctionne sur un flux Unicode et n'a pas du tout besoin d'utiliser un encodage de caractères, alors la [confiance](#) n'est *pas pertinente* .

*Certains algorithmes alimentent l'analyseur en ajoutant directement des caractères au [flux d'entrée](#) plutôt qu'en ajoutant des octets au [flux d'octets d'entrée](#) .*

### 13.2.3.1 Analyse avec un codage de caractères connu

Lorsque l'analyseur HTML doit fonctionner sur un flux d'octets d'entrée qui a un **codage défini connu** , alors le codage des caractères est ce codage et la [confiance](#) est *certaine* .

### 13.2.3.2 Détermination du codage des caractères

Dans certains cas, il peut être impossible de déterminer sans ambiguïté le codage avant d'analyser le document. Pour cette raison, cette spécification prévoit un mécanisme à deux passages avec un pré-balayage facultatif. Les implémentations sont autorisées, comme décrit ci-dessous, à appliquer un algorithme d'analyse simplifié à tous les octets dont elles disposent avant de commencer à analyser le document. Ensuite, le véritable analyseur est démarré, en utilisant un encodage provisoire dérivé de cette pré-analyse et d'autres métadonnées hors bande. Si, pendant le chargement du document, l'agent utilisateur découvre une déclaration d'encodage de caractères qui est en conflit avec ces informations, alors l'analyseur peut être réinvoqué pour effectuer une analyse du document avec l'encodage réel.

Les agents utilisateurs doivent utiliser l'algorithme suivant, appelé **algorithme de reniflage d'encodage** , pour déterminer l'encodage de caractères à utiliser lors du décodage d'un document lors de la première passe. Cet algorithme prend en entrée toutes les métadonnées hors bande disponibles pour l'agent utilisateur (par exemple, les [métadonnées Content-Type](#) du document) et tous les octets disponibles jusqu'à présent, et renvoie un codage de caractères et une *confiance* [provisoire](#) ou *certaine* . .

1. Si le résultat du [reniflage de nomenclature](#) est un encodage, renvoyez cet encodage avec [une confiance certaine](#) .

*Bien que l' algorithme [de décodage](#) modifie lui-même l'encodage à utiliser en fonction de la présence d'une marque d'ordre d'octet, cet algorithme renifle*

*également la nomenclature afin de définir l' encodage et [la confiance des caractères du document](#) correct .*

2. Si l'utilisateur a explicitement demandé à l'agent utilisateur de remplacer l'encodage de caractères du document par un encodage spécifique, renvoyez éventuellement cet encodage avec la [confiance](#) *certain* .

*En règle générale, les agents utilisateurs se souviennent de ces demandes d'utilisateurs à travers les sessions et, dans certains cas, les appliquent [iframe](#) également aux documents dans s.*

3. L'agent utilisateur peut attendre que d'autres octets de la ressource soient disponibles, soit à cette étape, soit à n'importe quelle étape ultérieure de cet algorithme. Par exemple, un agent utilisateur peut attendre 500 ms ou 1024 octets, selon la première éventualité. En général, la préparation de la source pour trouver l'encodage améliore les performances, car elle réduit la nécessité de jeter les structures de données utilisées lors de l'analyse lors de la recherche des informations d'encodage. Cependant, si l'agent utilisateur tarde trop longtemps à obtenir des données pour déterminer le codage, le coût du retard pourrait l'emporter sur toute amélioration des performances par rapport à la préparation.

*Les exigences de conformité de création pour les déclarations de codage de caractères les limitent à n'apparaître que [dans les 1024 premiers octets](#) . Les agents utilisateurs sont donc encouragés à utiliser l'algorithme de prés캔 ci-dessous (tel qu'invoqué par ces étapes) sur les 1024 premiers octets, mais à ne pas caler au-delà.*

4. Si la couche de transport spécifie un codage de caractères et qu'il est pris en charge, renvoyez ce codage avec la [confiance](#) *certain* .
5. Optionnellement , [pré-analyser le flux d'octets pour déterminer son encodage](#) , la [condition de fin](#) étant lorsque l'agent utilisateur décide que l'analyse d'autres octets ne serait pas efficace. Les agents utilisateurs sont encouragés à ne prés캔ner que les 1024 premiers octets. Les agents utilisateurs peuvent décider que l'analyse *de n'importe quel* octet n'est pas efficace, auquel cas ces sous-étapes sont entièrement sautées.

L'algorithme susmentionné renvoie soit un codage de caractères, soit un échec. S'il renvoie un codage de caractères, renvoie le même codage, avec [confiance](#) *provisoire* .

6. Si l' [analyseur HTML](#) pour lequel cet algorithme est exécuté est associé à un [Document](#) *d* dont [le document conteneur](#) est non nul, alors :
  1. Soit *parentDocument* le [document conteneur](#) de *d* .
  2. Si l'[origine](#) de *parentDocument* est [la même origine](#) que l'[origine](#) de *d* et que l'[encodage de caractères](#) de *parentDocument* n'est pas [UTF-16BE/LE](#) , alors renvoie l'[encodage de caractères](#) de *parentDocument* , avec la [confiance](#) *provisoire* .



7. Sinon, si l'agent utilisateur a des informations sur l'encodage probable de cette page, par exemple sur la base de l'encodage de la page lors de sa dernière visite, alors retournez cet encodage, avec la [confiance](#) provisoire .
8. L'agent utilisateur peut tenter de détecter automatiquement le codage de caractères en appliquant une analyse de fréquence ou d'autres algorithmes au flux de données. De tels algorithmes peuvent utiliser des informations sur la ressource autres que le contenu de la ressource, y compris l'adresse de la ressource. Si la détection automatique réussit à déterminer un encodage de caractères et que cet encodage est un encodage pris en charge, renvoyez cet encodage, avec la [confiance](#) provisoire . [\[UNIVCHARDET\]](#)

*Les agents utilisateurs sont généralement découragés de tenter de détecter automatiquement les codages pour les ressources obtenues sur le réseau, car cela implique par nature des heuristiques non interoperables. Tenter de détecter des encodages basés sur le préambule d'un document HTML est particulièrement délicat car le balisage HTML n'utilise généralement que des caractères ASCII, et les documents HTML ont tendance à commencer avec beaucoup de balisage plutôt qu'avec du contenu textuel. Le codage UTF-8 a un modèle de bits hautement détectable. Les fichiers du système de fichiers local qui contiennent des octets avec des valeurs supérieures à 0x7F qui correspondent au modèle UTF-8 sont très susceptibles d'être UTF-8, tandis que les documents avec des séquences d'octets qui ne lui correspondent pas ne le sont très probablement pas. Lorsqu'un agent utilisateur peut examiner l'intégralité du fichier, plutôt que le préambule uniquement, la détection spécifique d'UTF-8 peut être particulièrement efficace. [\[PPUTF8\]](#) [\[UTF8DET\]](#)*

9. Sinon, renvoie un codage de caractères par défaut [défini par l'implémentation](#) ou spécifié par l'utilisateur, avec la valeur [de confiance](#) provisoire .

Dans les environnements contrôlés ou dans les environnements où l'encodage des documents peut être prescrit (par exemple, pour les agents utilisateurs destinés à un usage dédié dans les nouveaux réseaux), l' UTF-8 encodage complet est suggéré.

Dans d'autres environnements, l'encodage par défaut dépend généralement des paramètres régionaux de l'utilisateur (une approximation des langues, et donc souvent des encodages, des pages que l'utilisateur est susceptible de fréquenter). Le tableau suivant donne des suggestions par défaut basées sur les paramètres régionaux de l'utilisateur, pour la compatibilité avec le contenu hérité. Les paramètres régionaux sont identifiés par les balises de langue BCP 47. [\[BCP47\]](#) [\[ENCODAGE\]](#)

Langue locale		Encodage par défaut suggéré
ar	arabe	<a href="#">fenêtres-1256</a>
az	Azéris	<a href="#">fenêtres-1254</a>
ba	Bachkir	<a href="#">fenêtres-1251</a>

Langue locale		Encodage par défaut suggéré
ête	biélorusse	<a href="#">fenêtres-1251</a>
bg	bulgare	<a href="#">fenêtres-1251</a>
cs	tchèque	<a href="#">fenêtres-1250</a>
el	grec	<a href="#">ISO-8859-7</a>
et	estonien	<a href="#">fenêtres-1257</a>
FA	persan	<a href="#">fenêtres-1256</a>
il	hébreu	<a href="#">fenêtres-1255</a>
heure	croate	<a href="#">fenêtres-1250</a>
heu	hongrois	<a href="#">ISO-8859-2</a>
ja	Japonais	<a href="#">Maj_JIS</a>
kk	Kazakh	<a href="#">fenêtres-1251</a>
ko	coréen	<a href="#">EUC-KR</a>
ku	kurde	<a href="#">fenêtres-1254</a>
ky	Kirghize	<a href="#">fenêtres-1251</a>
ça	lituanien	<a href="#">fenêtres-1257</a>
LV	letton	<a href="#">fenêtres-1257</a>
mk	Macédonien	<a href="#">fenêtres-1251</a>
PL	polonais	<a href="#">ISO-8859-2</a>
ru	russe	<a href="#">fenêtres-1251</a>
sah	Yakout	<a href="#">fenêtres-1251</a>
sk	slovaque	<a href="#">fenêtres-1250</a>
sl	slovène	<a href="#">ISO-8859-2</a>
sr	serbe	<a href="#">fenêtres-1251</a>
TG	tadjik	<a href="#">fenêtres-1251</a>
e	thaïlandais	<a href="#">fenêtres-874</a>
tr	turc	<a href="#">fenêtres-1254</a>
tt	tatar	<a href="#">fenêtres-1251</a>
Royaume-Uni	ukrainien	<a href="#">fenêtres-1251</a>
vi	vietnamien	<a href="#">fenêtres-1258</a>
zh-Hans, zh-CN, zh-SG	Chinois simplifié	<a href="#">GBK</a>
zh-Hant, zh-HK, zh-MO, zh-TW	Chinois (Traditionnel)	<a href="#">Big5</a>
Tous les autres paramètres régionaux		<a href="#">fenêtres-1252</a>

Le contenu de ce tableau est dérivé de l'intersection des valeurs par défaut de Windows, Chrome et Firefox.



Le [codage des caractères du document](#) doit immédiatement être défini sur la valeur renvoyée par cet algorithme, en même temps que l'agent utilisateur utilise la valeur renvoyée pour sélectionner le décodeur à utiliser pour le flux d'octets d'entrée.

---

Lorsqu'un algorithme demande à un agent utilisateur de **préscanner un flux d'octets pour déterminer son encodage**, compte tenu d'une **condition de fin** définie, il doit alors exécuter les étapes suivantes. Si à un moment quelconque au cours de ces étapes (y compris pendant les instances de l'algorithme [d'obtention d'un attribut](#) invoqué par celui-ci), l'agent utilisateur soit à court d'octets (ce qui signifie que le pointeur de *position* créé à la première étape ci-dessous va au-delà de la fin du flux d'octets obtenu jusqu'à présent) ou atteint sa *condition de fin*, puis interrompez le [pré-balayage d'un flux d'octets pour déterminer son](#) algorithme d'encodage et retournez le résultat [obtenir un encodage XML](#) appliqué aux mêmes octets que le [préscanner un flux d'octets pour déterminer son](#) algorithme de codage a été appliqué. Sinon, ces étapes renverront un codage de caractères.

1. Laissez *le codage de secours* être nul.
2. Soit *position* un pointeur vers un octet dans le flux d'octets d'entrée, pointant initialement sur le premier octet.
3. Analyse préalable des déclarations XML UTF-16 : si *la position* pointe vers :

**Une séquence d'octets commençant par : 0x3C, 0x0, 0x3F, 0x0, 0x78, 0x0 (sensible à la casse UTF-16 little-endian '<?x')**

Renvoie [UTF-16LE](#) .

**Une séquence d'octets commençant par : 0x0, 0x3C, 0x0, 0x3F, 0x0, 0x78 (big-endian UTF-16 sensible à la casse '<?x')**

Renvoie [UTF-16BE](#) .

*Pour des raisons historiques, le préfixe est plus long de deux octets que dans [l'annexe F](#) de XML et le nom d'encodage n'est pas vérifié.*

4. *Boucle* : Si *la position* pointe vers :

**Une séquence d'octets commençant par : 0x3C 0x21 0x2D 0x2D ( '<!--' )**

Avancez le pointeur de *position* de sorte qu'il pointe sur le premier octet 0x3E qui est précédé de deux octets 0x2D (c'est-à-dire à la fin d'une séquence ASCII '-->') et vient après l'octet 0x3C qui a été trouvé. (Les deux octets 0x2D peuvent être identiques à ceux de la séquence '<!--'.)

**Une séquence d'octets commençant par : 0x3C, 0x4D ou 0x6D, 0x45 ou 0x65, 0x54 ou 0x74, 0x41 ou 0x61, et l'un parmi 0x09, 0x0A, 0x0C, 0x0D,**

**0x20, 0x2F (ASCII insensible à la casse '<meta' suivi de un espace ou une barre oblique)**

1. Avancez le pointeur de *position* de manière à ce qu'il pointe sur l'octet suivant 0x09, 0x0A, 0x0C, 0x0D, 0x20 ou 0x2F (celui de la séquence de caractères correspondant ci-dessus).
2. Soit *la liste d'attributs* une liste vide de chaînes.
3. Que *pragma* soit faux.
4. Soit *need pragma* nul.
5. Soit *charset* la valeur nulle (qui, pour les besoins de cet algorithme, est distincte d'un encodage non reconnu ou de la chaîne vide).
6. *Attributs* : [Récupère un attribut](#) et sa valeur. Si aucun attribut n'a été renflé, passez à l'étape de *traitement* ci-dessous.
7. Si le nom de l'attribut est déjà dans *la liste des attributs* , revenez à l'étape intitulée *attributs* .
8. Ajoutez le nom de l'attribut à *la liste d'attributs* .
9. Exécutez l'étape appropriée dans la liste suivante, le cas échéant :

**Si le nom de l'attribut est " http-equiv "**

Si la valeur de l'attribut est " content-type ", alors définissez *got pragma* sur true.

**Si le nom de l'attribut est " content "**

Appliquez l' [algorithme pour extraire un encodage de caractères d'un meta-élément](#) , en donnant la valeur de l'attribut comme chaîne à analyser. Si un encodage de caractères est renvoyé et si *charset* est toujours défini sur null, laissez *charset* être l'encodage renvoyé et définissez *need pragma* sur true.

**Si le nom de l'attribut est " charset "**

Laissez *charset* être le résultat de [l'obtention d'un encodage](#) à partir de la valeur de l'attribut et définissez *need pragma* sur false.

10. Revenez à l'étape intitulée *attributs* .
11. *Traitement* : si *le pragma de besoin* est nul, passez à l'étape ci-dessous étiquetée *byte suivant* .
12. Si *le pragma de besoin* est vrai mais que *le pragma obtenu* est faux, passez à l'étape ci-dessous intitulée *byte suivant* .
13. Si *le jeu de caractères* est un échec, passez à l'étape ci-dessous intitulée *octet suivant* .
14. Si *charset* est [UTF-16BE/LE](#) , alors définissez *charset* sur [UTF-8](#) .

15. Si *charset* est [x-user-defined](#) , alors définissez *charset* sur [windows-1252](#) .

16. Retourner *le jeu de caractères* .

**Une séquence d'octets commençant par un octet 0x3C (<), éventuellement un octet 0x2F (/), et enfin un octet dans la plage 0x41-0x5A ou 0x61-0x7A (AZ ou az)**

17. Avancez le pointeur de *position* afin qu'il pointe sur l'octet 0x09 (HT), 0x0A (LF), 0x0C (FF), 0x0D (CR), 0x20 (SP) ou 0x3E (>) suivant.

18. [Obtenez](#) à plusieurs reprises un attribut jusqu'à ce qu'aucun autre attribut ne puisse être trouvé, puis passez à l'étape ci-dessous étiquetée *byte suivant* .

**Une séquence d'octets commençant par : 0x3C 0x21 ( ` <!` )**

**Une séquence d'octets commençant par : 0x3C 0x2F ( ` </` )**

**Une séquence d'octets commençant par : 0x3C 0x3F ( ` <?` )**

Avancez le pointeur de *position* afin qu'il pointe sur le premier octet 0x3E (>) qui vient après l'octet 0x3C qui a été trouvé.

#### **Tout autre octet**

Ne rien faire avec cet octet.

5. *Octet suivant* : déplacez *la position* de manière à ce qu'elle pointe sur l'octet suivant dans le flux d'octets d'entrée et revenez à l'étape ci-dessus intitulée *loop* .

Lorsque le [préscan d'un flux d'octets pour déterminer son](#) algorithme de codage indique d' **obtenir un attribut** , cela signifie faire ceci :

1. Si l'octet à *la position* est l'un des 0x09 (HT), 0x0A (LF), 0x0C (FF), 0x0D (CR), 0x20 (SP) ou 0x2F (/), alors avancez la position à l'octet suivant et refaites cette étape.
2. Si l'octet à *la position* est 0x3E (>), alors abandonnez l' algorithme [d'obtention d'un attribut](#) . Il n'y en a pas.
3. Sinon, l'octet à *la position* est le début du nom de l'attribut. Laissez *le nom d'attribut* et *la valeur d'attribut* être la chaîne vide.
4. Traiter l'octet à *la position* comme suit :

**S'il s'agit de 0x3D (=) et que le *nom de l'attribut* est plus long que la chaîne vide**

Avancez *la position* jusqu'à l'octet suivant et passez à l'étape sous *la valeur* étiquetée .

**S'il s'agit de 0x09 (HT), 0x0A (LF), 0x0C (FF), 0x0D (CR) ou 0x20 (SP)**

Passez à l'étape située sous *les espaces* étiquetés .

**Si c'est 0x2F (/) ou 0x3E (>)**

Abandonner l' algorithme [d'obtention d'un attribut](#) . Le nom de l'attribut est la valeur de *nom de l'attribut* , sa valeur est la chaîne vide.

**S'il est dans la plage 0x41 (A) à 0x5A (Z)**

Ajoutez le point de code  $b + 0x20$  au *nom de l'attribut* (où  $b$  est la valeur de l'octet à *la position* ). (Cela convertit l'entrée en minuscules.)

**Rien d'autre**

Ajoutez le point de code avec la même valeur que l'octet à *la position de l'attribut name* . (Peu importe la façon dont les octets en dehors de la plage ASCII sont traités ici, puisque seuls les octets ASCII peuvent contribuer à la détection d'un codage de caractères.)

5. Avancez *la position* jusqu'à l'octet suivant et revenez à l'étape précédente.
6. *Espaces* : si l'octet à *la position* est l'un des 0x09 (HT), 0x0A (LF), 0x0C (FF), 0x0D (CR) ou 0x20 (SP), avancez *la position* jusqu'à l'octet suivant, puis répétez cette étape.
7. Si l'octet à *la position* n'est pas 0x3D (=), abandonnez l' algorithme [d'obtention d'un attribut](#) . Le nom de l'attribut est la valeur de *nom de l'attribut* , sa valeur est la chaîne vide.
8. Avancez *la position* au-delà de l'octet 0x3D (=).
9. *Valeur* : si l'octet à *la position* est l'un des 0x09 (HT), 0x0A (LF), 0x0C (FF), 0x0D (CR) ou 0x20 (SP), avancez *la position* jusqu'à l'octet suivant, puis répétez cette étape.
10. Traiter l'octet à *la position* comme suit :

**Si c'est 0x22 (") ou 0x27 (')**

1. Soit  $b$  la valeur de l'octet en *position* .
2. *Boucle de citation* : Avance *la position* à l'octet suivant.
3. Si la valeur de l'octet à *la position* est la valeur de  $b$  , alors avancez *la position* à l'octet suivant et abandonnez l'algorithme "obtenir un attribut". Le nom de l'attribut est la valeur de *l'attribut name* , et sa valeur est la valeur de *l'attribut value* .
4. Sinon, si la valeur de l'octet à *la position* est comprise entre 0x41 (A) et 0x5A (Z), ajoutez un point de code à *la valeur d'attribut* dont la valeur est supérieure de 0x20 à la valeur de l'octet à *la position* .
5. Sinon, ajoutez un point de code à *la valeur d'attribut* dont la valeur est identique à la valeur de l'octet à *la position* .

6. Revenez à l'étape ci-dessus intitulée *boucle de citation* .

**Si c'est 0x3E (>)**

Abandonner l' algorithme [d'obtention d'un attribut](#) . Le nom de l'attribut est la valeur de *nom de l'attribut* , sa valeur est la chaîne vide.

**S'il est dans la plage 0x41 (A) à 0x5A (Z)**

Ajoutez un point de code  $b + 0x20$  à la valeur de l'attribut (où  $b$  est la valeur de l'octet à la position ). Avance la position à l'octet suivant.

**Rien d'autre**

Ajoutez un point de code avec la même valeur que l'octet en position à l'attribut *value* . Avance la position à l'octet suivant.

11. Traiter l'octet à la position comme suit :

**S'il s'agit de 0x09 (HT), 0x0A (LF), 0x0C (FF), 0x0D (CR), 0x20 (SP) ou 0x3E (>)**

Abandonner l' algorithme [d'obtention d'un attribut](#) . Le nom de l'attribut est la valeur de l'attribut *name* et sa valeur est la valeur de l'attribut *value* .

**S'il est dans la plage 0x41 (A) à 0x5A (Z)**

Ajoutez un point de code  $b + 0x20$  à la valeur de l'attribut (où  $b$  est la valeur de l'octet à la position ).

**Rien d'autre**

Ajoutez un point de code avec la même valeur que l'octet en position à l'attribut *value* .

12. Avancez la position jusqu'à l'octet suivant et revenez à l'étape précédente.

Lorsque le [pré-balayage d'un flux d'octets pour déterminer son](#) algorithme de codage est abandonné sans renvoyer de codage, **obtenir un codage XML** signifie le faire.

*La recherche d'une syntaxe ressemblant à une déclaration XML, même dans [text/html](#), est nécessaire pour la compatibilité avec le contenu existant.*

1. Soit *encodingPosition* un pointeur vers le début du flux.
2. Si *encodingPosition* ne pointe pas vers le début d'une séquence d'octets 0x3C, 0x3F, 0x78, 0x6D, 0x6C ( `<?xml`` ), alors renvoie un échec.
3. Soit *xmlDeclarationEnd* un pointeur vers l'octet suivant dans le flux d'octets d'entrée qui est 0x3E (>). S'il n'y a pas un tel octet, alors renvoyez l'échec.
4. Définissez *encodingPosition* sur la position de la première occurrence de la sous-séquence d'octets 0x65, 0x6E, 0x63, 0x6F, 0x64, 0x69, 0x6E, 0x67 ( `` `` ) à *encoding* ou après le *encodingPosition* actuel . S'il n'y a pas une telle séquence, alors renvoyez l'échec.
5. Avancez *encodingPosition* après l'octet 0x67 (g).

6. Tant que l'octet à *encodingPosition* est inférieur ou égal à 0x20 (c'est-à-dire qu'il s'agit soit d'un espace ASCII, soit d'un caractère de contrôle), avancez *encodingPosition* à l'octet suivant.
7. Si l'octet à *encodingPosition* n'est pas 0x3D (=), alors renvoie un échec.
8. Avancez *encodingPosition* jusqu'à l'octet suivant.
9. Tant que l'octet à *encodingPosition* est inférieur ou égal à 0x20 (c'est-à-dire qu'il s'agit soit d'un espace ASCII, soit d'un caractère de contrôle), avancez *encodingPosition* à l'octet suivant.
10. Soit *quoteMark* l'octet à *encodingPosition* .
11. Si *quoteMark* n'est ni 0x22 (") ni 0x27 ('), alors renvoie un échec.
12. Avancez *encodingPosition* jusqu'à l'octet suivant.
13. Soit *encodingEndPosition* la position de la prochaine occurrence de *quoteMark* à ou après *encodingPosition* . Si *quoteMark* ne se reproduit pas, renvoie l'échec.
14. Soit *potentialEncoding* la séquence d'octets entre *encodingPosition* (inclusif) et *encodingEndPosition* (exclusif).
15. Si *potentialEncoding* contient un ou plusieurs octets dont la valeur est inférieure ou égale à 0x20, renvoie un échec.
16. Soit *encoding* le résultat de [l'obtention d'un encodage](#) donné *potentialEncoding* [isomorphique décodé](#) .
17. Si l' *encodage* est [UTF-16BE/LE](#) , changez-le en [UTF-8](#) .
18. *Encodage* de retour .

Pour des raisons d'interopérabilité, les agents utilisateurs ne devraient pas utiliser un algorithme de pré-analyse qui renvoie des résultats différents de celui décrit ci-dessus. (Mais, si vous le faites, veuillez au moins nous le faire savoir, afin que nous puissions améliorer cet algorithme et que tout le monde en profite...)

### 13.2.3.3 Encodages de caractères

Les agents utilisateurs doivent prendre en charge les encodages définis dans *Encoding* , y compris, mais sans s'y limiter, [UTF-8](#) , [ISO-8859-2](#) , [ISO-8859-7](#) , [ISO-8859-8](#) , [windows-874](#) , [windows-1250](#) , [windows-1251](#) , [fenêtres-1252](#) , [fenêtres-1254](#) , [fenêtres-1255](#) , [fenêtres-1256](#) , [fenêtres-1257](#) , [fenêtres-1258](#) , [GBK](#) , [Big5](#) , [ISO-2022-JP](#) , [Shift JIS](#) , [EUC-KR](#) , [UTF-16BE](#) , [UTF-16LE](#) , [UTF-16BE/LE](#) et [x-user-defined](#) . Les agents utilisateurs ne doivent pas prendre en charge d'autres encodages.



*Ce qui précède interdit de prendre en charge, par exemple, CESU-8, UTF-7, BOCU-1, SCSU, EBCDIC et UTF-32. Cette spécification ne fait aucune tentative pour prendre en charge les codages interdits dans ses algorithmes ; la prise en charge et l'utilisation d'encodages interdits conduiraient donc à des comportements inattendus. [\[CESU8\]](#) [\[UTF7\]](#) [\[BOCU1\]](#) [\[SCSU\]](#)*

#### 13.2.3.4 Modification de l'encodage lors de l'analyse

Lorsque l'analyseur demande à l'agent utilisateur de **modifier l'encodage**, il doit exécuter les étapes suivantes. Cela peut se produire si l' [algorithme de reniflage d'encodage](#) décrit ci-dessus n'a pas trouvé d'encodage de caractères, ou s'il a trouvé un encodage de caractères qui n'était pas l'encodage réel du fichier.

1. Si l'encodage déjà utilisé pour interpréter le flux d'entrée est [UTF-16BE/LE](#), définissez la [confiance](#) sur *certain* et revenez. Le nouveau codage est ignoré ; si c'était autre chose que le même encodage, alors ce serait clairement incorrect.
2. Si le nouveau codage est [UTF-16BE/LE](#), changez-le en [UTF-8](#).
3. Si le nouvel encodage est [x-user-defined](#), remplacez-le par [windows-1252](#).
4. Si le nouveau codage est identique ou équivalent au codage déjà utilisé pour interpréter le flux d'entrée, définissez la [confiance](#) sur *certain* et retournez. Cela se produit lorsque les informations d'encodage trouvées dans le fichier correspondent à ce que l' [algorithme de reniflage d'encodage](#) a déterminé comme étant l'encodage, et lors de la deuxième passe à travers l'analyseur si la première passe a trouvé que l'algorithme de reniflage d'encodage décrit dans la section précédente n'a pas réussi à trouver le bon codage.
5. Si tous les octets jusqu'au dernier octet convertis par le décodeur actuel ont les mêmes interprétations Unicode dans le codage actuel et le nouveau codage, et si l'agent utilisateur prend en charge le changement de convertisseur à la volée, alors l'agent utilisateur peut passer au nouveau convertisseur pour l'encodage à la volée. Définissez l' [encodage des caractères du document](#) et l'encodage utilisé pour convertir le flux d'entrée vers le nouvel encodage, définissez la [confiance](#) sur *certain* et retournez.
6. Sinon, redémarrez l'algorithme [de navigation](#), avec [historyHandling](#) défini sur `"replace"` et les autres entrées conservées, mais cette fois ignorez l' [algorithme de reniflage d'encodage](#) et définissez simplement l'encodage sur le nouvel encodage et la [confiance](#) sur *certain*. Dans la mesure du possible, cela devrait être fait sans réellement contacter la couche réseau (les octets devraient être ré-analysés à partir de la mémoire), même si, par exemple, le document est marqué comme n'étant pas cachable. Si cela n'est pas possible et que contacter la couche réseau impliquerait de répéter une demande qui utilise une méthode autre que `GET`, alors définissez plutôt

la [confiance](#) sur *certain*set ignorez le nouvel encodage. La ressource sera mal interprétée. Les agents utilisateurs peuvent informer l'utilisateur de la situation, pour aider au développement de l'application.

*Cet algorithme n'est invoqué que lorsqu'un nouvel encodage est trouvé déclaré sur un [meta](#)élément.*

### 13.2.3.5 Prétraitement du flux d'entrée

Le **flux d'entrée** se compose des caractères qui y sont insérés lors du décodage du [flux d'octets d'entrée](#) ou des différentes API qui manipulent directement le flux d'entrée.

Toutes les occurrences de [substituts](#) sont [des erreurs d'analyse de substitution dans le flux d'entrée](#) . Toutes les occurrences de [non-caractères](#) sont [des erreurs d'analyse non-caractères dans le flux d'entrée](#) et toutes les occurrences de [contrôles](#) autres que [les espaces blancs ASCII](#) et les caractères U+0000 NULL sont [des erreurs d'analyse de caractères de contrôle dans le flux d'entrée](#) .

*La gestion des caractères U+0000 NULL varie en fonction de l'endroit où les caractères sont trouvés et se produit aux étapes ultérieures de l'analyse. Ils sont soit ignorés soit, pour des raisons de sécurité, remplacés par un CARACTERE DE REMPLACEMENT U+FFFD. Cette gestion est, par nécessité, répartie à la fois sur l'étape de tokenisation et sur l'étape de construction de l'arborescence.*

Avant l' étape [de tokenisation](#) , le flux d'entrée doit être prétraité en [normalisant les retours à la ligne](#) . Ainsi, les retours à la ligne dans les DOM HTML sont représentés par des caractères U+000A LF, et il n'y a jamais de caractères U+000D CR dans l'entrée de l' étape [de tokenisation](#) .

Le **caractère d'entrée suivant** est le premier caractère du [flux d'entrée](#) qui n'a pas encore été **consommé** ou explicitement ignoré par les exigences de cette section. Initialement, le [caractère d'entrée suivant](#) est le premier caractère de l'entrée. Le **caractère d'entrée actuel** est le dernier caractère à avoir été *consommé* .

Le **point d'insertion** est la position (juste avant un caractère ou juste avant la fin du flux d'entrée) où le contenu inséré à l'aide `document.write()` est réellement inséré. Le point d'insertion est relatif à la position du caractère immédiatement après, il ne s'agit pas d'un décalage absolu dans le flux d'entrée. Initialement, le point d'insertion n'est pas défini.

Le caractère "EOF" dans les tableaux ci-dessous est un caractère conceptuel représentant la fin du [flux d'entrée](#) . Si l'analyseur est un [analyseur créé par un script](#) , la fin du [flux d'entrée](#) est atteinte lorsqu'un **caractère "EOF" explicite** (inséré par la `document.close()` méthode) est utilisé. Sinon, le caractère "EOF" n'est pas un vrai caractère dans le flux, mais plutôt l'absence de tout autre caractère.



## 13.2.4 État d'analyse

### 13.2.4.1 Le mode d'insertion

Le **mode d'insertion** est une variable d'état qui contrôle l'opération principale de l'étape de construction de l'arbre.

Initialement, le [mode d'insertion](#) est « [initial](#) ». Il peut changer en " [avant html](#) ", " [avant la tête](#) ", " [dans la tête](#) ", " [dans la tête noscript](#) ", " [après la tête](#) ", " [dans le corps](#) ", " [texte](#) ", " [dans le tableau](#) ", " [dans le texte du tableau](#) ", " [dans la légende](#) ", " [dans le groupe de colonnes](#) ", " [dans le corps du tableau](#) ", " [dans la ligne](#) ", " [dans la cellule](#) ", " [dans la sélection](#) ", "", " [in frameset](#) ", " [after frameset](#) ", " [after after body](#) " et " [after after frameset](#) " au cours de l'analyse, comme décrit dans l'étape [de construction de l'arborescence](#) . Le mode d'insertion affecte la façon dont les jetons sont traités et si CDATA les rubriques sont prises en charge.

Plusieurs de ces modes, à savoir " [in head](#) ", " [in body](#) ", " [in table](#) ", et " [in select](#) ", sont particuliers, en ce que les autres modes s'y reportent à des instants divers. Lorsque l'algorithme ci-dessous indique que l'agent utilisateur doit faire quelque chose " **en utilisant les règles du mode d'insertion  $m$**  ", où  $m$  est l'un de ces modes, l'agent utilisateur doit utiliser les règles décrites dans la section du [mode d'insertion  \$m\$](#)  , mais doit laisser le [mode d'insertion](#) inchangé à moins que les règles de  $m$  elles-mêmes ne basculent le [mode d'insertion](#) vers une nouvelle valeur.

Lorsque le mode d'insertion est commuté sur « [texte](#) » ou « [dans le texte du tableau](#) », le **mode d'insertion d'origine** est également défini. C'est le mode d'insertion auquel reviendra l'étape de construction de l'arbre.

De même, pour analyser [template](#) les éléments imbriqués, une **pile de modes d'insertion de modèles** est utilisée. Il est initialement vide. Le **mode d'insertion de modèle actuel** est le mode d'insertion qui a été ajouté le plus récemment à la [pile des modes d'insertion de modèle](#) . Les algorithmes dans les sections ci-dessous pousseront les modes d'insertion sur cette pile, ce qui signifie que le mode d'insertion spécifié doit être ajouté à la pile, et les modes d'insertion *pop* de la pile, ce qui signifie que le mode d'insertion ajouté le plus récemment doit être supprimé de la pile. empiler.

---

Lorsque les étapes ci-dessous nécessitent que l'UA **réinitialise le mode d'insertion de manière appropriée** , cela signifie que l'UA doit suivre ces étapes :

1. Que *la dernière* soit fausse.
2. Soit *node* le dernier nœud de la [pile d'éléments ouverts](#) .

3. *Boucle* : si le *nœud* est le premier nœud de la pile d'éléments ouverts, définissez le *dernier* sur vrai et, si l'analyseur a été créé dans le cadre de l' [algorithme d'analyse de fragment HTML](#) ( [fragment case](#) ), définissez le *nœud* sur l' élément [de contexte](#) transmis à ce algorithme.
4. Si *node* est un [select](#) élément, exécutez ces sous-étapes :
  1. Si *last* est vrai, passez à l'étape ci-dessous intitulée *done* .
  2. Soit *ancêtre* *node* . \_
  3. *Boucle* : si l'*ancêtre* est le premier nœud de la [pile d'éléments ouverts](#) , passez à l'étape ci-dessous étiquetée *terminée* .
  4. Soit *ancêtre* le nœud avant *ancêtre* dans la [pile des éléments ouverts](#) .
  5. Si l'*ancêtre* est un [template](#) nœud, passez à l'étape ci-dessous intitulée *done* .
  6. Si l'*ancêtre* est un [table](#) nœud, basculez le [mode d'insertion](#) sur " [in select in table](#) " et revenez.
  7. Revenez à l'étape intitulée *loop* .
  8. *Terminé* : Basculez le [mode d'insertion](#) sur " [dans sélectionner](#) " et revenez.
5. Si *node* est un élément [td](#) ou et *last* est faux, basculez le [mode d'insertion](#) sur " [in cell](#) " et revenez. [th](#)
6. Si *node* est un [tr](#) élément, basculez le [mode d'insertion](#) sur " [in row](#) " et revenez.
7. Si le *nœud* est un élément [tbody](#), [thead](#) ou [tfoot](#), basculez le [mode d'insertion](#) sur « [dans le corps du tableau](#) » et revenez.
8. Si *node* est un [caption](#) élément, basculez le [mode d'insertion](#) sur " [in caption](#) " et revenez.
9. Si le *nœud* est un [colgroup](#) élément, basculez le [mode d'insertion](#) sur " [dans le groupe de colonnes](#) " et revenez.
10. Si *node* est un [table](#) élément, basculez le [mode d'insertion](#) sur " [in table](#) " et revenez.
11. Si le *nœud* est un [template](#) élément, basculez le [mode d'insertion](#) sur le [mode d'insertion de modèle actuel](#) et revenez.
12. Si *node* est un [head](#) élément et *last* vaut false, alors basculez le [mode d'insertion](#) sur " [in head](#) " et revenez.

13. Si le *nœud* est un bodyélément, basculez le mode d'insertion sur " dans le corps " et revenez.
14. Si *node* est un framesetélément, basculez le mode d'insertion sur " in frameset " et revenez. ( cas de fragment )
15. Si *node* est un htmlélément, exécutez ces sous-étapes :
  1. Si le headpointeur d'élément est nul, basculez le mode d'insertion sur « avant tête » et revenez. ( cas de fragment )
  2. Sinon, le headpointeur d'élément n'est pas nul, passez le mode d'insertion à " after head " et revenez.
16. Si *last* est vrai, basculez le mode d'insertion sur " in body " et revenez. ( cas de fragment )
17. Soit *node* maintenant le nœud avant *node* dans la pile des éléments ouverts .
18. Revenez à l'étape intitulée *loop* .

#### 13.2.4.2 L'empilement des éléments ouverts

Initialement, la **pile d'éléments ouverts** est vide. La pile grandit vers le bas; le nœud le plus haut de la pile est le premier ajouté à la pile, et le nœud le plus bas de la pile est le nœud ajouté le plus récemment dans la pile (nonobstant lorsque la pile est manipulée de manière aléatoire dans le cadre de la gestion des erreurs mal imbriquées balises ).

*Le mode d'insertion " avant html " crée l' élément document , qui est ensuite ajouté à la pile.html  
Dans le cas du fragment , la pile d'éléments ouverts est initialisée pour contenir un htmlélément créé dans le cadre de cet algorithme . (La cas du fragment ignore le mode d'insertion " avant html " .)*

Le htmlnœud, quelle que soit sa création, est le nœud le plus haut de la pile. Il n'est retiré de la pile que lorsque l'analyseur se termine .

Le **nœud courant** est le nœud le plus bas de cette pile d'éléments ouverts .

Le **nœud actuel ajusté** est l' élément de contexte si l'analyseur a été créé dans le cadre de l' algorithme d'analyse de fragment HTML et que la pile d'éléments ouverts ne contient qu'un seul élément ( cas de fragment ) ; sinon, le nœud courant ajusté est le nœud courant .

Les éléments de la pile d'éléments ouverts appartiennent aux catégories suivantes :

## Spécial

Les éléments suivants ont différents niveaux de règles d'analyse spéciales :  
HTML [address](#), [applet](#), [area](#), [article](#), [aside](#), [base](#), [basefont](#), [bgsound](#), [blockquote](#), [body](#), [br](#), [button](#), [caption](#), [center](#), [col](#), [colgroup](#), [dd](#), [details](#), [dir](#), [div](#), [dl](#), [dt](#), [embed](#), [fieldset](#), [figcaption](#), [figure](#), [footer](#), [form](#), [frame](#), [frameset](#), [h1](#), [h2](#), [h3](#), [h4](#), [h5](#), [h6](#), [head](#), [header](#), [hgroup](#), [hr](#), [html](#), [iframe](#), [img](#), [input](#), [keygen](#), [link](#), [listing](#), [main](#), [marquee](#), [menu](#), [meta](#), [nav](#), [noembed](#), [noframes](#), [noscript](#), [object](#), [ol](#), [p](#), [param](#), [plaintext](#), [pre](#), [script](#), [section](#), [select](#), [source](#), [style](#), [summary](#), [table](#), [tbody](#), [td](#), [template](#), [textarea](#), [tfoot](#), [th](#), [thead](#), [title](#), [tr](#), [track](#), [math](#), [mathml](#), [mathml](#), [mathml](#), [mathml](#), [mathml](#) et [mathml](#) ; et [SVG](#), [SVG](#) et [ulwbrxmpmimomnmsmttextannotation-xmlforeignObjectdescSVG title](#).

Un [image](#) jeton de balise de début est géré par le générateur d'arbre, mais il n'est pas dans cette liste car ce n'est pas un élément ; il devient un [img](#) élément.

## Mise en page

Les éléments HTML suivants sont ceux qui se retrouvent dans la [liste des éléments de formatage actifs](#) : [a](#), [b](#), [big](#), [code](#), [em](#), [font](#), [i](#), [nobr](#), [s](#), [small](#), [strike](#), [strong](#), [tt](#), et [u](#).

## Ordinaire

Tous les autres éléments trouvés lors de l'analyse d'un document HTML.

En règle générale, les éléments [spéciaux](#) ont les jetons de balise de début et de fin gérés spécifiquement, tandis que les jetons des éléments [ordinaires](#) relèvent des clauses "toute autre balise de début" et "toute autre balise de fin", et certaines parties du générateur d'arbre vérifient si un élément particulier dans la [pile d'éléments ouverts](#) est dans la catégorie [spéciale](#). Cependant, certains éléments (par exemple, l'[option](#) élément) ont leurs jetons de balise de début ou de fin gérés spécifiquement, mais ne sont toujours pas dans la catégorie [spéciale](#), de sorte qu'ils obtiennent le traitement [ordinaire](#) ailleurs.

On dit que la [pile d'éléments ouverts](#) a un **nœud cible d'élément** dans une portée **spécifique** consistant en une liste de types d'éléments lorsque l'algorithme suivant se termine dans un état de correspondance :

1. Initialiser le *nœud* pour qu'il soit le [nœud actuel](#) (le nœud le plus bas de la pile).
2. Si le *nœud* est le nœud cible, terminer dans un état de correspondance.
3. Sinon, si *node* est l'un des types d'éléments de *list*, se termine par un état d'échec.
4. Sinon, définissez *node* sur l'entrée précédente dans la [pile d'éléments ouverts](#) et revenez à l'étape 2. (Cela n'échouera jamais, car la boucle se terminera toujours à l'étape précédente si le sommet de la pile - un [html](#) élément - est atteint.)

La [pile d'éléments ouverts](#) est dite **avoir un élément particulier dans la portée** lorsqu'elle [a cet élément dans la portée spécifique](#) composée des types d'éléments suivants :

- [applet](#)
- [caption](#)
- [html](#)
- [table](#)
- [td](#)
- [th](#)
- [marquee](#)
- [object](#)
- [template](#)
- [MathML<sub>mi</sub>](#)
- [MathML<sub>mo</sub>](#)
- [MathML<sub>mn</sub>](#)
- [MathML<sub>ms</sub>](#)
- [MathML<sub>mtext</sub>](#)
- [MathML<sub>annotation-xml</sub>](#)
- [SVG<sub>foreignObject</sub>](#)
- [SVG<sub>desc</sub>](#)
- [SVG<sub>title</sub>](#)

La [pile d'éléments ouverts](#) est dite **avoir un élément particulier dans la portée de l'élément de liste** lorsqu'elle [a cet élément dans la portée spécifique](#) composée des types d'éléments suivants :

- Tous les types d'éléments répertoriés ci-dessus pour le [ont un élément dans](#) l'algorithme de portée.
- [ol](#) dans l' [espace de noms HTML](#)
- [ul](#) dans l' [espace de noms HTML](#)

La [pile d'éléments ouverts](#) est dite **avoir un élément particulier dans la portée du bouton** lorsqu'elle [a cet élément dans la portée spécifique](#) composée des types d'éléments suivants :

- Tous les types d'éléments répertoriés ci-dessus pour le [ont un élément dans](#) l'algorithme de portée.
- [button](#) dans l' [espace de noms HTML](#)

La [pile d'éléments ouverts](#) est dite **avoir un élément particulier dans la portée de la table** lorsqu'elle [a cet élément dans la portée spécifique](#) composée des types d'éléments suivants :

- [html](#) dans l' [espace de noms HTML](#)
- [table](#) dans l' [espace de noms HTML](#)
- [template](#) dans l' [espace de noms HTML](#)

La [pile d'éléments ouverts](#) est dite **avoir un élément particulier dans la portée de sélection** lorsqu'elle [a cet élément dans la portée spécifique](#) composée de tous les types d'éléments à l'exception des suivants :

- [optgroup](#) dans l' [espace de noms HTML](#)
- [option](#) dans l' [espace de noms HTML](#)

Rien ne se passe si, à tout moment, l'un des éléments de la [pile d'éléments ouverts](#) est déplacé vers un nouvel emplacement ou supprimé de l' [Document](#) arborescence. En particulier, la pile n'est pas modifiée dans cette situation. Cela peut entraîner, entre autres effets étranges, l'ajout de contenu à des nœuds qui ne sont plus dans le DOM.

*Dans certains cas (à savoir, lors [de la fermeture d'éléments de formatage mal imbriqués](#) ), la pile est manipulée de manière aléatoire.*

#### 13.2.4.3 La liste des éléments de formatage actifs

Initialement, la **liste des éléments de formatage actifs** est vide. [Il est utilisé pour gérer les balises d'élément de formatage](#) mal imbriquées .

La liste contient des éléments de la catégorie [de mise en forme](#) et [des marqueurs](#) . Les **marqueurs** sont insérés lors de la saisie des éléments [applet](#), [object](#), [marquee](#), [template](#), [td](#), [th](#), et [caption](#), et sont utilisés pour empêcher la mise en forme de "fuir" *dans* les éléments [applet](#), [object](#), [marquee](#), [template](#), [td](#), [th](#), et [caption](#)

De plus, chaque élément de la [liste des éléments de formatage actifs](#) est associé au jeton pour lequel il a été créé, de sorte que d'autres éléments peuvent être créés pour ce jeton si nécessaire.

Lorsque les étapes ci-dessous demandent à l'UA de **pousser sur la liste des éléments de formatage actifs** un élément *element* , l'UA doit effectuer les étapes suivantes :

1. S'il y a déjà trois éléments dans la [liste des éléments de mise en forme actifs](#) après le dernier [marqueur](#) , le cas échéant, ou n'importe où dans la liste s'il n'y a pas de [marqueurs](#) , qui ont le même nom de balise, espace de noms et attributs que *element* , supprimez le plus ancien cet élément dans la [liste des éléments de formatage actifs](#) . À ces fins, les attributs doivent être comparés tels qu'ils étaient lorsque les éléments ont été créés par l'analyseur ; deux éléments ont les mêmes attributs si tous leurs attributs analysés peuvent être appariés de sorte que les deux attributs de chaque paire aient des noms, des espaces de noms et des valeurs identiques (l'ordre des attributs n'a pas d'importance).

*C'est la clause de l'Arche de Noé. Mais à trois par famille au lieu de deux.*

2. Ajouter *un élément* à la [liste des éléments de formatage actifs](#) .

Lorsque les étapes ci-dessous nécessitent que l'UA **reconstruise les éléments de formatage actifs** , l'UA doit effectuer les étapes suivantes :

1. S'il n'y a pas d'entrées dans la [liste des éléments de formatage actifs](#) , alors il n'y a rien à reconstruire ; arrêter cet algorithme.
2. Si la dernière entrée (la plus récemment ajoutée) dans la [liste des éléments de formatage actifs](#) est un [marqueur](#) , ou s'il s'agit d'un élément qui se trouve dans la [pile des éléments ouverts](#) , alors il n'y a rien à reconstruire ; arrêter cet algorithme.
3. Soit *entry* le dernier élément (ajouté le plus récemment) dans la [liste des éléments de formatage actifs](#) .
4. *Rembobiner* : s'il n'y a pas d'entrées avant *l'entrée* dans la [liste des éléments de formatage actifs](#) , passez à l'étape intitulée *créer* .
5. Soit *entrée* l'entrée une avant *entrée* dans la [liste des éléments de formatage actifs](#) .
6. Si *entry* n'est ni un [marqueur](#) ni un élément qui se trouve également dans la [pile d'éléments ouverts](#) , passez à l'étape intitulée *rewind* .
7. *Avance* : Soit *entrée* l'élément un après *l'entrée* dans la [liste des éléments de formatage actifs](#) .
8. *Créer* : [Insérez un élément HTML](#) pour le jeton pour lequel l' *entrée* d'élément a été créée, afin d'obtenir *un nouvel élément* .
9. Remplacez l'entrée pour *entrée* dans la liste par une entrée pour *nouvel élément* .
10. Si l'entrée pour *nouvel élément* dans la [liste des éléments de formatage actifs](#) n'est pas la dernière entrée de la liste, retournez à l'étape intitulée *avance* .

Cela a pour effet de rouvrir tous les éléments de mise en forme qui ont été ouverts dans le corps, la cellule ou la légende actuelle (selon la plus récente) qui n'ont pas été explicitement fermés.

*La façon dont cette spécification est écrite, la [liste des éléments de formatage actifs](#) se compose toujours d'éléments dans l'ordre chronologique avec l'élément le moins récemment ajouté en premier et l'élément le plus récemment ajouté en dernier (sauf pendant que les étapes 7 à 10 de l'algorithme ci-dessus sont en cours d'exécution, bien sûr).*

Lorsque les étapes ci-dessous nécessitent que l'UA **efface la liste des éléments de formatage actifs jusqu'au dernier marqueur** , l'UA doit effectuer les étapes suivantes :



1. Soit *entrée* la dernière entrée (la plus récemment ajoutée) dans la [liste des éléments de formatage actifs](#) .
2. Supprimer *l'entrée* de la [liste des éléments de formatage actifs](#) .
3. Si *entry* était un [marqueur](#) , arrêtez l'algorithme à ce stade. La liste a été effacée jusqu'au dernier [repère](#) .
4. Passez à l'étape 1.

#### 13.2.4.4 Les pointeurs d'éléments

Initialement, le **head**pointeur d'élément et le **form**pointeur d'élément sont tous les deux nuls.

Une fois qu'un **head**élément a été analysé (implicitement ou explicitement), le [head](#)pointeur d'élément est défini pour pointer vers ce nœud.

Le [form](#)pointeur d'élément pointe sur le dernier **form**élément qui a été ouvert et dont la balise de fin n'a pas encore été vue. Il est utilisé pour associer les contrôles de formulaire aux formulaires face à un balisage dramatiquement mauvais, pour des raisons historiques. Il est ignoré à l'intérieur [template](#) des éléments.

#### 13.2.4.5 Autres drapeaux d'état d'analyse

L' **indicateur de script** est défini sur "activé" si [le script a été activé](#) pour le [Document](#) auquel l'analyseur est associé lors de la création de l'analyseur, et sur "désactivé" dans le cas contraire.

*L' [indicateur de script](#) peut être activé même lorsque l'analyseur a été créé dans le cadre de l' [algorithme d'analyse de fragment HTML](#) , même si [script](#) les éléments ne s'exécutent pas dans ce cas.*

L' **indicateur frameset-ok** est défini sur "ok" lorsque l'analyseur est créé. Il est défini sur "pas ok" après que certains jetons sont vus.

#### 13.2.5 Tokénisation

Les implémentations doivent agir comme si elles utilisaient la machine d'état suivante pour tokeniser HTML. La machine d'état doit démarrer dans l' [état de données](#) . La plupart des états consomment un seul caractère, ce qui peut avoir divers effets



secondaires, et soit fait passer la machine d'état à un nouvel état pour [reprendre](#) le [caractère d'entrée actuel](#) , soit la fait passer à un nouvel état pour consommer le [caractère suivant](#) , soit reste dans le même état pour consommer le caractère suivant. Certains états ont un comportement plus compliqué et peuvent consommer plusieurs caractères avant de passer à un autre état. Dans certains cas, l'état du tokenizer est également modifié par l'étape de construction de l'arborescence.

Lorsqu'un état dit de **reconsommer** un caractère correspondant dans un état spécifié, cela signifie de passer à cet état, mais lorsqu'il tente de consommer le [caractère d'entrée suivant](#) , fournissez-lui le [caractère d'entrée actuel](#) à la place.

Le comportement exact de certains états dépend du [mode d'insertion](#) et de la [pile d'éléments ouverts](#) . Certains états utilisent également un **tampon temporaire** pour suivre la progression, et l' [état de référence de caractère](#) utilise un **état de retour** pour revenir à l'état à partir duquel il a été invoqué.

La sortie de l'étape de tokenisation est une série de zéro ou plusieurs des jetons suivants : DOCTYPE, balise de début, balise de fin, commentaire, caractère, fin de fichier. Les jetons DOCTYPE ont un nom, un identifiant public, un identifiant système et un **indicateur force-quirks** . Lorsqu'un jeton DOCTYPE est créé, son nom, son identifiant public et son identifiant système doivent être marqués comme manquants (ce qui est un état distinct de la chaîne vide), et l'indicateur [force-quirks](#) doit être désactivé (son autre état est *activé* ). Les jetons de balise de début et de fin ont un nom de balise, un **indicateur de fermeture automatique** et une liste d'attributs, chacun ayant un nom et une valeur. Lorsqu'un jeton de balise de début ou de fin est créé, son [indicateur de fermeture automatique](#) doit être désactivé (son autre état est défini) et sa liste d'attributs doit être vide. Les jetons de commentaire et de caractère contiennent des données.

Lorsqu'un jeton est émis, il doit immédiatement être pris en charge par l' étape [de construction de l'arbre](#) . L'étape de construction de l'arborescence peut affecter l'état de l'étape de tokenisation et peut insérer des caractères supplémentaires dans le flux. (Par exemple, l' [script](#) élément peut entraîner l'exécution de scripts et l'utilisation des API [d'insertion de balisage dynamique](#) pour insérer des caractères dans le flux en cours de tokenisation.)

*Créer un jeton et l'émettre sont des actions distinctes. Il est possible qu'un jeton soit créé mais implicitement abandonné (jamais émis), par exemple si le fichier se termine de manière inattendue lors du traitement des caractères qui sont analysés dans un jeton de balise de début.*

Lorsqu'un jeton de balise de début est émis avec son [indicateur de fermeture automatique](#) défini, si l'indicateur n'est pas **reconnu** lorsqu'il est traité par l'étape de construction de l'arbre, c'est un [non-void-html-element-start-tag-with-trailing- erreur d'analyse solidus](#) .

Lorsqu'un jeton de balise de fin est émis avec des attributs, il s'agit d'une [erreur d'analyse de balise de fin avec attributs](#) .

Lorsqu'un jeton de balise de fin est émis avec son [indicateur de fermeture automatique](#) défini, il s'agit d'une [erreur d'analyse end-tag-with-trailing-solidus](#) .

Un **jeton de balise de fin approprié** est un jeton de balise de fin dont le nom de balise correspond au nom de balise de la dernière balise de début émise par ce tokenizer, le cas échéant. Si aucune balise de début n'a été émise par ce tokenizer, aucun jeton de balise de fin n'est approprié.

Une [référence de caractère](#) est dite consommée **dans le cadre d'un attribut** si l' [état de retour](#) est soit [la valeur d'attribut \(entre guillemets\) state](#) , [la valeur d'attribut \(entre guillemets simples\) state](#) ou [la valeur d'attribut \(sans guillemets\) state](#) .

Lorsqu'un état indique de **vider les points de code consommés en tant que référence de caractère** , cela signifie que pour chaque [point de code](#) dans le [tampon temporaire](#) (dans l'ordre dans lequel ils ont été ajoutés au tampon), l'agent utilisateur doit ajouter le point de code du tampon à l'attribut actuel. value si la référence de caractère a été [consommée dans le cadre d'un attribut](#) , ou émettre le point de code en tant que jeton de caractère dans le cas contraire.

Avant chaque étape du tokenizer, l'agent utilisateur doit d'abord vérifier l' [indicateur de pause de l'analyseur](#) . Si c'est vrai, alors le tokenizer doit abandonner le traitement de toutes les invocations imbriquées du tokenizer, rendant le contrôle à l'appelant.

La machine d'état du tokenizer se compose des états définis dans les sous-sections suivantes.

#### 13.2.5.1 *État des données*

Consomme le [caractère d'entrée suivant](#) :

##### **U+0026 AMPERSAND (&)**

Définissez l' [état de retour](#) sur l' [état des données](#) . Passez à l' [état de référence de caractère](#) .

##### **U+003C SIGNE INFÉRIEUR À (<)**

Passez à l' [état ouvert de la balise](#) .

##### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

##### **EOF**

Émettre un jeton de fin de fichier.

##### **Rien d'autre**

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.2 *État RCDATA*

Consomme le [caractère d'entrée suivant](#) :

#### **U+0026 AMPERSAND (&)**

Définissez l' [état de retour](#) sur l' [état RCDATA](#) . Passez à l' [état de référence de caractère](#) .

#### **U+003C SIGNE INFÉRIEUR À (<)**

Passez à l' [état de signe inférieur à RCDATA](#) .

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### **EOF**

Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.3 *Etat TEXTE BRUT*

Consomme le [caractère d'entrée suivant](#) :

#### **U+003C SIGNE INFÉRIEUR À (<)**

Passez à l' [état RAWTEXT signe inférieur à](#) .

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### **EOF**

Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.4 *État des données de script*

Consomme le [caractère d'entrée suivant](#) :

#### **U+003C SIGNE INFÉRIEUR À (<)**

Passez à l'[état des données de script inférieur à signe](#) .

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### **EOF**

Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### **13.2.5.5 Etat TEXTE CLAIR**

Consomme le [caractère d'entrée suivant](#) :

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### **EOF**

Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### **13.2.5.6 Etat ouvert de l'étiquette**

Consomme le [caractère d'entrée suivant](#) :

#### **U+0021 POINT D'EXCLAMATION (!)**

Passez à [l'état ouvert de la déclaration de balisage](#) .

#### **U+002F SOLIDE (/)**

Passez à l' [état ouvert de la balise de fin](#) .

#### **Alpha ASCII**

Créez un nouveau jeton de balise de début, définissez son nom de balise sur la chaîne vide. [Reprendre](#) dans l' [état du nom de balise](#) .

#### **U+003F POINT D'INTERROGATION (?)**

Il s'agit d'une [erreur d'analyse inattendue de point d'interrogation au lieu de nom de balise](#) . Créez un jeton de commentaire dont les données sont la chaîne vide. [Reprendre](#) dans l' [état de faux commentaire](#) .

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-before-tag-name](#) . Émettre un jeton de caractère U+003C LESS-THAN SIGN et un jeton de fin de fichier.

#### Rien d'autre

Il s'agit d'une [erreur d'analyse du premier caractère du nom de balise non valide](#) . Émettez un jeton de caractère U+003C SIGNE MOINS QUE. [Reconsommer](#) dans l' [état de données](#) .

### 13.2.5.7 *Etat ouvert d'étiquette de fin*

Consomme le [caractère d'entrée suivant](#) :

#### Alpha ASCII

Créez un nouveau jeton de balise de fin, définissez son nom de balise sur la chaîne vide. [Reprendre](#) dans l' [état du nom de balise](#) .

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse de nom de balise de fin manquante](#) . Passez à l' [état des données](#) .

#### EOF

Il s'agit d'une [erreur d'analyse eof-before-tag-name](#) . Émettez un jeton de caractère U+003C LESS-THAN SIGN, un jeton de caractère U+002F SOLIDUS et un jeton de fin de fichier.

#### Rien d'autre

Il s'agit d'une [erreur d'analyse du premier caractère du nom de balise non valide](#) . Créez un jeton de commentaire dont les données sont la chaîne vide. [Reprendre](#) dans l' [état de faux commentaire](#) .

### 13.2.5.8 *État du nom de variable*

Consomme le [caractère d'entrée suivant](#) :

#### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

#### **U+000A ALIMENTATION LIGNE (LF)**

#### **U+000C SAUT DE FORMULAIRE (FF)**

#### **U+0020 ESPACE**

Passez à l' [état avant le nom d'attribut](#) .

#### **U+002F SOLIDE (/)**

Passez à l' [état de balise de début à fermeture automatique](#) .

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton de balise actuel.

### **Alpha supérieur ASCII**

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au nom de balise du jeton de balise actuel.

### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de remplacement U+FFFD au nom de balise du jeton de balise actuel.

### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-tag](#) . Émettre un jeton de fin de fichier.

### **Rien d'autre**

Ajoute le [caractère d'entrée actuel](#) au nom de balise du jeton de balise actuel.

## **13.2.5.9 État de signe inférieur à RCDATA**

Consomme le [caractère d'entrée suivant](#) :

### **U+002F SOLIDE (/)**

Définissez le [tampon temporaire](#) sur la chaîne vide. Passez à l' [état ouvert de balise de fin RCDATA](#) .

### **Rien d'autre**

Émettez un jeton de caractère U+003C SIGNE MOINS QUE. [Reprendre](#) dans l' [état RCDATA](#) .

## **13.2.5.10 État ouvert de l'étiquette de fin RCDATA**

Consomme le [caractère d'entrée suivant](#) :

### **Alpha ASCII**

Créez un nouveau jeton de balise de fin, définissez son nom de balise sur la chaîne vide. [Reprendre](#) dans l' [état du nom de balise de fin RCDATA](#) .

### **Rien d'autre**

Émettez un jeton de caractère U+003C LESS-THAN SIGN et un jeton de caractère U+002F SOLIDUS. [Reprendre](#) dans l' [état RCDATA](#) .

## **13.2.5.11 État du nom de balise de fin RCDATA**

Consomme le [caractère d'entrée suivant](#) :

#### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

#### **U+000A ALIMENTATION LIGNE (LF)**

#### **U+000C SAUT DE FORMULAIRE (FF)**

#### **U+0020 ESPACE**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état avant le nom d' attribut](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### **U+002F SOLIDE (/)**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de balise de début à fermeture automatique](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de données](#) et émettez le jeton de balise actuel. Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### **Alpha supérieur ASCII**

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

#### **Alpha inférieur ASCII**

Ajoute le [caractère d'entrée actuel](#) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

#### **Rien d'autre**

Émettez un jeton de caractère U+003C LESS-THAN SIGN, un jeton de caractère U+002F SOLIDUS et un jeton de caractère pour chacun des caractères dans le [tampon temporaire](#) (dans l'ordre dans lequel ils ont été ajoutés au tampon). [Reprendre](#) dans l' [état RCDATA](#) .

### **13.2.5.12 RAWTEXT état de signe inférieur à**

Consomme le [caractère d'entrée suivant](#) :

#### **U+002F SOLIDE (/)**

Définissez le [tampon temporaire](#) sur la chaîne vide. Passez à l' [état ouvert de la balise de fin RAWTEXT](#) .

#### **Rien d'autre**

Émettez un jeton de caractère U+003C SIGNE MOINS QUE. [Reprendre](#) à l' [état RAWTEXT](#) .

### 13.2.5.13 État ouvert de la balise de fin RAWTEXT

Consomme le [caractère d'entrée suivant](#) :

#### Alpha ASCII

Créez un nouveau jeton de balise de fin, définissez son nom de balise sur la chaîne vide. [Reprendre](#) dans l' [état du nom de la balise de fin RAWTEXT](#) .

#### Rien d'autre

Émettez un jeton de caractère U+003C LESS-THAN SIGN et un jeton de caractère U+002F SOLIDUS. [Reprendre](#) à l' [état RAWTEXT](#) .

### 13.2.5.14 État du nom de la balise de fin RAWTEXT

Consomme le [caractère d'entrée suivant](#) :

#### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

#### **U+000A ALIMENTATION LIGNE (LF)**

#### **U+000C SAUT DE FORMULAIRE (FF)**

#### **U+0020 ESPACE**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état avant le nom d' attribut](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### **U+002F SOLIDE (/)**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de balise de début à fermeture automatique](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de données](#) et émettez le jeton de balise actuel. Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### Alpha supérieur ASCII

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

#### Alpha inférieur ASCII

Ajoute le [caractère d'entrée actuel](#) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

#### Rien d'autre

Émettez un jeton de caractère U+003C LESS-THAN SIGN, un jeton de caractère U+002F SOLIDUS et un jeton de caractère pour chacun des



caractères dans le [tampon temporaire](#) (dans l'ordre dans lequel ils ont été ajoutés au tampon). [Reprendre](#) à l' [état RAWTEXT](#) .

#### **13.2.5.15 Etat des données de script inférieur à signe**

Consomme le [caractère d'entrée suivant](#) :

##### **U+002F SOLIDE (/)**

Définissez le [tampon temporaire](#) sur la chaîne vide. Passez à [l'état ouvert de balise de fin de données de script](#) .

##### **U+0021 POINT D'EXCLAMATION (!)**

Passez à [l'état de démarrage de l'échappement des données de script](#) . Émettez un jeton de caractère U+003C LESS-THAN SIGN et un jeton de caractère U+0021 EXCLAMATION MARK.

##### **Rien d'autre**

Émettez un jeton de caractère U+003C SIGNE MOINS QUE. [Reprendre](#) dans l' [état des données de script](#) .

#### **13.2.5.16 État ouvert d'étiquette de fin de données de script**

Consomme le [caractère d'entrée suivant](#) :

##### **[Alpha ASCII](#)**

Créez un nouveau jeton de balise de fin, définissez son nom de balise sur la chaîne vide. [Reconsommer](#) dans l' [état du nom de balise de fin de données de script](#) .

##### **Rien d'autre**

Émettez un jeton de caractère U+003C LESS-THAN SIGN et un jeton de caractère U+002F SOLIDUS. [Reprendre](#) dans l' [état des données de script](#) .

#### **13.2.5.17 État du nom de balise de fin de données de script**

Consomme le [caractère d'entrée suivant](#) :

##### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

##### **U+000A ALIMENTATION LIGNE (LF)**

##### **U+000C SAUT DE FORMULAIRE (FF)**

## U+0020 ESPACE

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état avant le nom d' attribut](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

## U+002F SOLIDE (/)

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de balise de début à fermeture automatique](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

## U+003E SIGNE SUPÉRIEUR À (>)

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de données](#) et émettez le jeton de balise actuel. Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

### Alpha supérieur ASCII

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

### Alpha inférieur ASCII

Ajoute le [caractère d'entrée actuel](#) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

### Rien d'autre

Émettez un jeton de caractère U+003C LESS-THAN SIGN, un jeton de caractère U+002F SOLIDUS et un jeton de caractère pour chacun des caractères dans le [tampon temporaire](#) (dans l'ordre dans lequel ils ont été ajoutés au tampon). [Reprendre](#) dans l' [état des données de script](#) .

## 13.2.5.18 *Etat de début d'échappement des données de script*

Consomme le [caractère d'entrée suivant](#) :

## U+002D TIRET-MOINS (-)

Passez à l'[état du tiret de début d'échappement des données de script](#) . Émettez un jeton de caractère U+002D HYPHEN-MINUS.

### Rien d'autre

[Reprendre](#) dans l' [état des données de script](#) .

## 13.2.5.19 *État du tiret de début d'échappement des données de script*

Consomme le [caractère d'entrée suivant](#) :

#### **U+002D TIRET-MOINS (-)**

Passez à [l'état du tiret échappé des données de script](#) . Émettez un jeton de caractère U+002D HYPHEN-MINUS.

#### **Rien d'autre**

[Reprendre](#) dans l' [état des données de script](#) .

### **13.2.5.20 État d'échappement des données de script**

Consomme le [caractère d'entrée suivant](#) :

#### **U+002D TIRET-MOINS (-)**

Passez à [l'état du tiret échappé des données de script](#) . Émettez un jeton de caractère U+002D HYPHEN-MINUS.

#### **U+003C SIGNE INFÉRIEUR À (<)**

Passez aux [données de script échappées à l'état inférieur au signe](#) .

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-script-html-comment-like-text](#) . Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### **13.2.5.21 État du tiret échappé des données de script**

Consomme le [caractère d'entrée suivant](#) :

#### **U+002D TIRET-MOINS (-)**

Passez à [l'état du tiret échappé des données de script](#) . Émettez un jeton de caractère U+002D HYPHEN-MINUS.

#### **U+003C SIGNE INFÉRIEUR À (<)**

Passez aux [données de script échappées à l'état inférieur au signe](#) .

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Passez à l' [état d'échappement des données de script](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

## EOF

Il s'agit d'une [erreur d'analyse eof-in-script-html-comment-like-text](#) . Émettre un jeton de fin de fichier.

## Rien d'autre

Passez à l' [état d'échappement des données de script](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.22 Données de script échappées tiret état tiret

Consomme le [caractère d'entrée suivant](#) :

#### U+002D TIRET-MOINS (-)

Émettez un jeton de caractère U+002D HYPHEN-MINUS.

#### U+003C SIGNE INFÉRIEUR À (<)

Passez aux [données de script échappées à l'état inférieur au signe](#) .

#### U+003E SIGNE SUPÉRIEUR À (>)

Passez à l' [état des données de script](#) . Émettez un jeton de caractère U+003E SIGNE SUPÉRIEUR À.

#### U+0000 NUL

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Passez à l' [état d'échappement des données de script](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

## EOF

Il s'agit d'une [erreur d'analyse eof-in-script-html-comment-like-text](#) . Émettre un jeton de fin de fichier.

## Rien d'autre

Passez à l' [état d'échappement des données de script](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.23 Les données de script ont échappé à l'état inférieur à signe

Consomme le [caractère d'entrée suivant](#) :

#### U+002F SOLIDE (/)

Définissez le [tampon temporaire](#) sur la chaîne vide. Passez à [l'état ouvert de la balise de fin échappée des données de script](#) .

## [Alpha ASCII](#)

Définissez le [tampon temporaire](#) sur la chaîne vide. Émettez un jeton de caractère U+003C SIGNE MOINS QUE. [Reconsommer](#) dans le [script data double escape start state](#) .

#### Rien d'autre

Émettez un jeton de caractère U+003C SIGNE MOINS QUE. [Reconsommer](#) dans l' [état échappé des données de script](#) .

### 13.2.5.24 *Etat ouvert de balise de fin échappée de données de script*

Consomme le [caractère d'entrée suivant](#) :

#### Alpha ASCII

Créez un nouveau jeton de balise de fin, définissez son nom de balise sur la chaîne vide. [Reconsommer](#) dans le [script les données échappées de l'état du nom de la balise de fin](#) .

#### Rien d'autre

Émettez un jeton de caractère U+003C LESS-THAN SIGN et un jeton de caractère U+002F SOLIDUS. [Reconsommer](#) dans l' [état échappé des données de script](#) .

### 13.2.5.25 *Etat du nom de la balise de fin échappée des données de script*

Consomme le [caractère d'entrée suivant](#) :

#### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

#### **U+000A ALIMENTATION LIGNE (LF)**

#### **U+000C SAUT DE FORMULAIRE (FF)**

#### **U+0020 ESPACE**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état avant le nom d' attribut](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### **U+002F SOLIDE (/)**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de balise de début à fermeture automatique](#) . Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Si le jeton de balise de fin actuel est un [jeton de balise de fin approprié](#) , passez à l' [état de données](#) et émettez le jeton de balise actuel. Sinon, traitez-le comme indiqué dans l'entrée "autre chose" ci-dessous.

### Alpha supérieur ASCII

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

### Alpha inférieur ASCII

Ajoute le [caractère d'entrée actuel](#) au nom de balise du jeton de balise actuel. Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) .

### Rien d'autre

Émettez un jeton de caractère U+003C LESS-THAN SIGN, un jeton de caractère U+002F SOLIDUS et un jeton de caractère pour chacun des caractères dans le [tampon temporaire](#) (dans l'ordre dans lequel ils ont été ajoutés au tampon). [Reconsommer](#) dans l' [état échappé des données de script](#) .

## **13.2.5.26 Etat de début d'échappement double des données de script**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

**U+002F SOLIDE (/)**

**U+003E SIGNE SUPÉRIEUR À (>)**

Si le [tampon temporaire](#) est la chaîne " `script`", passez à l' [état d'échappement double des données de script](#) . Sinon, passez à l' [état échappé des données de script](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### Alpha supérieur ASCII

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au [buffer temporaire](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### Alpha inférieur ASCII

Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### Rien d'autre

[Reconsommer](#) dans l' [état échappé des données de script](#) .

### 13.2.5.27 État d'échappement double des données de script

Consomme le [caractère d'entrée suivant](#) :

#### U+002D TIRET-MOINS (-)

Passez à [l'état du tiret à double échappement des données de script](#) . Émettez un jeton de caractère U+002D HYPHEN-MINUS.

#### U+003C SIGNE INFÉRIEUR À (<)

Basculez vers les [données de script à double échappement état inférieur à signe](#) . Émettez un jeton de caractère U+003C SIGNE MOINS QUE.

#### U+0000 NUL

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### EOF

Il s'agit d'une [erreur d'analyse eof-in-script-html-comment-like-text](#) . Émettre un jeton de fin de fichier.

#### Rien d'autre

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.28 État du tiret à double échappement des données de script

Consomme le [caractère d'entrée suivant](#) :

#### U+002D TIRET-MOINS (-)

Passez à [l'état du tiret à double échappement des données de script](#) . Émettez un jeton de caractère U+002D HYPHEN-MINUS.

#### U+003C SIGNE INFÉRIEUR À (<)

Basculez vers les [données de script à double échappement état inférieur à signe](#) . Émettez un jeton de caractère U+003C SIGNE MOINS QUE.

#### U+0000 NUL

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Passez à l' [état d'échappement double des données de script](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### EOF

Il s'agit d'une [erreur d'analyse eof-in-script-html-comment-like-text](#) . Émettre un jeton de fin de fichier.

#### Rien d'autre

Passez à l' [état d'échappement double des données de script](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.29 *Données de script tiret à double échappement état tiret*

Consomme le [caractère d'entrée suivant](#) :

#### **U+002D TIRET-MOINS (-)**

Émettez un jeton de caractère U+002D HYPHEN-MINUS.

#### **U+003C SIGNE INFÉRIEUR À (<)**

Basculez vers les [données de script à double échappement état inférieur à signe](#) . Émettez un jeton de caractère U+003C SIGNE MOINS QUE.

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données de script](#) . Émettez un jeton de caractère U+003E SIGNE SUPÉRIEUR À.

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Passez à l' [état d'échappement double des données de script](#) . Émettez un jeton de personnage U+FFFD REPLACEMENT CHARACTER.

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-script-html-comment-like-text](#) . Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Passez à l' [état d'échappement double des données de script](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### 13.2.5.30 *Les données de script ont échappé à deux fois à l'état de signe inférieur à*

Consomme le [caractère d'entrée suivant](#) :

#### **U+002F SOLIDE (/)**

Définissez le [tampon temporaire](#) sur la chaîne vide. Passez à l' [état final d'échappement double des données de script](#) . Émettez un jeton de caractère U+002F SOLIDUS.

#### **Rien d'autre**

[Reconsommer](#) dans l' [état d'échappement double des données de script](#) .

### 13.2.5.31 *État final d'échappement double des données de script*

Consomme le [caractère d'entrée suivant](#) :



**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

**U+002F SOLIDE (/)**

**U+003E SIGNE SUPÉRIEUR À (>)**

Si le [tampon temporaire](#) est la chaîne " `script`", passez à l' [état échappé des données de script](#) . Sinon, passez à l' [état d'échappement double des données de script](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### **Alpha supérieur ASCII**

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au [buffer temporaire](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

### **Alpha inférieur ASCII**

Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) . Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

**Rien d'autre**

[Reconsommer](#) dans l' [état d'échappement double des données de script](#) .

### **13.2.5.32 Avant l'état du nom d'attribut**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Ignorez le personnage.

**U+002F SOLIDE (/)**

**U+003E SIGNE SUPÉRIEUR À (>)**

**EOF**

[Reprendre](#) dans l' [état après le nom d'attribut](#) .

**U+003D SIGNE ÉGAL (=)**

Il s'agit d'une [erreur d'analyse inattendue du signe égal avant le nom de l'attribut](#) . Commencez un nouvel attribut dans le jeton de balise actuel. Définissez le nom de cet attribut sur le [caractère d'entrée actuel](#) et sa valeur sur la chaîne vide. Passez à l' [état du nom d'attribut](#) .

**Rien d'autre**

Commencez un nouvel attribut dans le jeton de balise actuel. Définissez ce nom d'attribut et cette valeur sur la chaîne vide. [Reprendre](#) dans l' [état du nom d'attribut](#) .

#### 13.2.5.33 *État du nom d'attribut*

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

**U+002F SOLIDE (/)**

**U+003E SIGNE SUPÉRIEUR À (>)**

**EOF**

[Reprendre](#) dans l' [état après le nom d'attribut](#) .

**U+003D SIGNE ÉGAL (=)**

Passez à l' [état de la valeur d'attribut avant](#) .

#### **Alpha supérieur ASCII**

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au nom de l'attribut actuel.

**U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de REMPLACEMENT U+FFFD au nom de l'attribut actuel.

**U+0022 GUILLEMET (")**

**U+0027 APOSTROPHE (')**

**U+003C SIGNE INFÉRIEUR À (<)**

Il s'agit d'une [erreur d'analyse inattendue de caractère dans le nom d'attribut](#) . Traitez-le comme dans l'entrée "autre chose" ci-dessous.

#### **Rien d'autre**

Ajoute le [caractère d'entrée actuel](#) au nom de l'attribut actuel.

Lorsque l'agent utilisateur quitte l'état du nom d'attribut (et avant d'émettre le jeton de balise, le cas échéant), le nom complet de l'attribut doit être comparé aux autres attributs du même jeton ; s'il existe déjà un attribut sur le jeton portant exactement le même nom, il s'agit d'une [erreur d'analyse d'attribut en double](#) et le nouvel attribut doit être supprimé du jeton.

*Si un attribut est ainsi supprimé d'un jeton, celui-ci et la valeur qui lui est associée, le cas échéant, ne sont jamais utilisés par la suite par l'analyseur et sont donc*

*effectivement supprimés. Toutefois, la suppression de l'attribut de cette manière ne modifie pas son statut en tant qu'"attribut actuel" pour les besoins du tokenizer.*

#### **13.2.5.34 Après l'état du nom d'attribut**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Ignorez le personnage.

**U+002F SOLIDE (/)**

Passez à l' [état de balise de début à fermeture automatique](#) .

**U+003D SIGNE ÉGAL (=)**

Passez à l' [état de la valeur d'attribut avant](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton de balise actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-tag](#) . Émettre un jeton de fin de fichier.

**Rien d'autre**

Commencez un nouvel attribut dans le jeton de balise actuel. Définissez ce nom d'attribut et cette valeur sur la chaîne vide. [Reprendre](#) dans l' [état du nom d'attribut](#) .

#### **13.2.5.35 Avant l'état de la valeur d'attribut**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Ignorez le personnage.

**U+0022 GUILLEMET (")**

Passez à l' [état de valeur d'attribut \(entre guillemets\)](#) .

#### **U+0027 APOSTROPHE (')**

Passez à l' [état de valeur d'attribut \(guillemets simples\)](#) .

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse de valeur d'attribut manquante](#) . Passez à l' [état des données](#) . Émettez le jeton de balise actuel.

#### **Rien d'autre**

[Reconsommer](#) dans l' [état de valeur d'attribut \(sans guillemets\)](#) .

### **13.2.5.36 État de la valeur d'attribut (entre guillemets)**

Consomme le [caractère d'entrée suivant](#) :

#### **U+0022 GUILLEMET (")**

Passez à l' [état après la valeur d'attribut \(quote\)](#) .

#### **U+0026 AMPERSAND (&)**

Définissez l' [état de retour](#) sur l' [état de la valeur d'attribut \(entre guillemets\)](#) . Passez à l' [état de référence de caractère](#) .

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de REMPLACEMENT U+FFFD à la valeur de l'attribut actuel.

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-tag](#) . Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Ajoute le [caractère d'entrée actuel](#) à la valeur de l'attribut actuel.

### **13.2.5.37 État de la valeur d'attribut (entre guillemets simples)**

Consomme le [caractère d'entrée suivant](#) :

#### **U+0027 APOSTROPHE (')**

Passez à l' [état après la valeur d'attribut \(quote\)](#) .

#### **U+0026 AMPERSAND (&)**

Définissez l' [état de retour](#) sur l' [état de la valeur d'attribut \(entre guillemets simples\)](#) . Passez à l' [état de référence de caractère](#) .

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de REMPLACEMENT U+FFFD à la valeur de l'attribut actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-tag](#) . Émettre un jeton de fin de fichier.

**Rien d'autre**

Ajoute le [caractère d'entrée actuel](#) à la valeur de l'attribut actuel.

### **13.2.5.38 État de valeur d'attribut (sans guillemets)**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Passez à l' [état avant le nom d'attribut](#) .

**U+0026 AMPERSAND (&)**

Définissez l' [état de retour](#) sur l' [état de la valeur d'attribut \(sans guillemets\)](#) . Passez à l' [état de référence de caractère](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton de balise actuel.

**U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de REMPLACEMENT U+FFFD à la valeur de l'attribut actuel.

**U+0022 GUILLEMET (")**

**U+0027 APOSTROPHE (')**

**U+003C SIGNE INFÉRIEUR À (<)**

**U+003D SIGNE ÉGAL (=)**

**U+0060 ACCENT GRAVE (`)**

Il s'agit d'une [erreur d'analyse de caractère inattendu dans la valeur d'attribut sans guillemets](#) . Traitez-le comme dans l'entrée "autre chose" ci-dessous.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-tag](#) . Émettre un jeton de fin de fichier.

**Rien d'autre**

Ajoute le [caractère d'entrée actuel](#) à la valeur de l'attribut actuel.

#### 13.2.5.39 *Après l'état de valeur d'attribut (entre guillemets)*

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Passez à l' [état avant le nom d'attribut](#) .

**U+002F SOLIDE (/)**

Passez à l' [état de balise de début à fermeture automatique](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton de balise actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-tag](#) . Émettre un jeton de fin de fichier.

**Rien d'autre**

Il s'agit d'une [erreur d'analyse d'espaces manquants entre les attributs](#) . [Reprendre](#) dans l' [état avant le nom d'attribut](#) .

#### 13.2.5.40 *État de la balise de début à fermeture automatique*

Consomme le [caractère d'entrée suivant](#) :

**U+003E SIGNE SUPÉRIEUR À (>)**

Définissez l' [indicateur de fermeture automatique](#) du jeton de balise actuel. Passez à l' [état des données](#) . Émettez le jeton de balise actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-tag](#) . Émettre un jeton de fin de fichier.

**Rien d'autre**

Il s'agit d'une [erreur d'analyse inattendue de solidus-in-tag](#) . [Reprendre](#) dans l' [état avant le nom d'attribut](#) .

#### 13.2.5.41 *État de commentaire erroné*

Consomme le [caractère d'entrée suivant](#) :

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettre le jeton de commentaire actuel.

#### EOF

Émettez le commentaire. Émettre un jeton de fin de fichier.

#### U+0000 NUL

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de remplacement U+FFFD aux données du jeton de commentaire.

#### Rien d'autre

Ajoute le [caractère d'entrée actuel](#) aux données du jeton de commentaire.

### 13.2.5.42 État ouvert de la déclaration de balisage

Si les quelques caractères suivants sont :

#### Deux caractères U+002D TIRET-MOINS (-)

Consommez ces deux caractères, créez un jeton de commentaire dont les données sont la chaîne vide et passez à l' [état de début de commentaire](#) .

#### [Correspondance ASCII non sensible à la casse](#) pour le mot "DOCTYPE"

Consommez ces caractères et passez à l' [état DOCTYPE](#) .

#### La chaîne "[CDATA[" (les cinq lettres majuscules "CDATA" avec un caractère U+005B LEFT SQUARE BRACKET avant et après)

Consommez ces personnages. S'il existe un [nœud actuel ajusté](#) et qu'il ne s'agit pas d'un élément dans l' [espace de noms HTML](#) , passez à l' [état de la section CDATA](#) . Sinon, il s'agit d'une [erreur d'analyse cdata-in-html-content](#) . Créez un jeton de commentaire dont les données sont la chaîne "[CDATA[" . Passez à l' [état de faux commentaire](#) .

#### Rien d'autre

Il s'agit d'une [erreur d'analyse de commentaire mal ouvert](#) . Créez un jeton de commentaire dont les données sont la chaîne vide. Passez à l' [état de faux commentaire](#) (ne consommez rien dans l'état actuel).

### 13.2.5.43 État de début de commentaire

Consomme le [caractère d'entrée suivant](#) :

#### U+002D TIRET-MOINS (-)

Passez à l' [état du tiret de début de commentaire](#) .

#### U+003E SIGNE SUPÉRIEUR À (>)

Il s'agit d'une [erreur d'analyse de la fermeture brutale d'un commentaire vide](#) . Passez à l' [état des données](#) . Émettre le jeton de commentaire actuel.

**Rien d'autre**

[Reprendre](#) dans l' [état de commentaire](#) .

#### **13.2.5.44 État du tiret de début de commentaire**

Consomme le [caractère d'entrée suivant](#) :

**U+002D TIRET-MOINS (-)**

Passez à l' [état de fin de commentaire](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse de la fermeture brutale d'un commentaire vide](#) . Passez à l' [état des données](#) . Émettre le jeton de commentaire actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-comment](#) . Émettre le jeton de commentaire actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**

Ajoutez un caractère U+002D HYPHEN-MINUS (-) aux données du jeton de commentaire. [Reprendre](#) dans l' [état de commentaire](#) .

#### **13.2.5.45 État des commentaires**

Consomme le [caractère d'entrée suivant](#) :

**U+003C SIGNE INFÉRIEUR À (<)**

Ajoute le [caractère d'entrée actuel](#) aux données du jeton de commentaire. Passez à l' [état de commentaire inférieur à signe](#) .

**U+002D TIRET-MOINS (-)**

Passez à l' [état du tiret de fin de commentaire](#) .

**U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de remplacement U+FFFD aux données du jeton de commentaire.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-comment](#) . Émettre le jeton de commentaire actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**



Ajoute le [caractère d'entrée actuel](#) aux données du jeton de commentaire.

#### **13.2.5.46 *Commentaire état inférieur à signe***

Consomme le [caractère d'entrée suivant](#) :

##### **U+0021 POINT D'EXCLAMATION (!)**

Ajoute le [caractère d'entrée actuel](#) aux données du jeton de commentaire. Passez à l' [état de bang de signe inférieur au commentaire](#) .

##### **U+003C SIGNE INFÉRIEUR À (<)**

Ajoute le [caractère d'entrée actuel](#) aux données du jeton de commentaire.

Rien d'autre

[Reprendre](#) dans l' [état de commentaire](#) .

#### **13.2.5.47 *Commentaire état de bang signe inférieur à***

Consomme le [caractère d'entrée suivant](#) :

##### **U+002D TIRET-MOINS (-)**

Passez au [commentaire inférieur à signe bang dash state](#) .

Rien d'autre

[Reprendre](#) dans l' [état de commentaire](#) .

#### **13.2.5.48 *Commentaire inférieur à signe bang état tiret***

Consomme le [caractère d'entrée suivant](#) :

##### **U+002D TIRET-MOINS (-)**

Passez au [commentaire signe inférieur à bang dash dash state](#) .

Rien d'autre

[Reprendre](#) dans l' [état de tiret de fin de commentaire](#) .

#### **13.2.5.49 *Commentaire signe inférieur à bang tiret tiret état***

Consomme le [caractère d'entrée suivant](#) :

#### **U+003E SIGNE SUPÉRIEUR À (>)**

**EOF**

[Reprendre](#) dans l' [état de fin de commentaire](#) .

**Rien d'autre**

Il s'agit d'une [erreur d'analyse de commentaires imbriqués](#) . [Reprendre](#) dans l' [état de fin de commentaire](#) .

#### **13.2.5.50 État du tiret de fin de commentaire**

Consomme le [caractère d'entrée suivant](#) :

#### **U+002D TIRET-MOINS (-)**

Passez à l' [état de fin de commentaire](#) .

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-comment](#) . Émettre le jeton de commentaire actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**

Ajoutez un caractère U+002D HYPHEN-MINUS (-) aux données du jeton de commentaire. [Reprendre](#) dans l' [état de commentaire](#) .

#### **13.2.5.51 État final du commentaire**

Consomme le [caractère d'entrée suivant](#) :

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettre le jeton de commentaire actuel.

#### **U+0021 POINT D'EXCLAMATION (!)**

Passez à l' [état de bang de fin de commentaire](#) .

#### **U+002D TIRET-MOINS (-)**

Ajoutez un caractère U+002D HYPHEN-MINUS (-) aux données du jeton de commentaire.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-comment](#) . Émettre le jeton de commentaire actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**

Ajoutez deux caractères U+002D HYPHEN-MINUS (-) aux données du jeton de commentaire. [Reprendre](#) dans l' [état de commentaire](#) .

#### 13.2.5.52 *État du coup de fin de commentaire*

Consomme le [caractère d'entrée suivant](#) :

##### **U+002D TIRET-MOINS (-)**

Ajoutez deux caractères U+002D HYPHEN-MINUS (-) et un caractère U+0021 EXCLAMATION MARK (!) aux données du jeton de commentaire. Passez à l' [état du tiret de fin de commentaire](#) .

##### **U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse de commentaire mal fermé](#) . Passez à l' [état des données](#) . Émettre le jeton de commentaire actuel.

##### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-comment](#) . Émettre le jeton de commentaire actuel. Émettre un jeton de fin de fichier.

##### **Rien d'autre**

Ajoutez deux caractères U+002D HYPHEN-MINUS (-) et un caractère U+0021 EXCLAMATION MARK (!) aux données du jeton de commentaire. [Reprendre](#) dans l' [état de commentaire](#) .

#### 13.2.5.53 *État DOCTYPE*

Consomme le [caractère d'entrée suivant](#) :

##### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

##### **U+000A ALIMENTATION LIGNE (LF)**

##### **U+000C SAUT DE FORMULAIRE (FF)**

##### **U+0020 ESPACE**

Passez à l' [état avant le nom DOCTYPE](#) .

##### **U+003E SIGNE SUPÉRIEUR À (>)**

[Reprendre](#) dans l' [état avant le nom DOCTYPE](#) .

##### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . Créez un nouveau jeton DOCTYPE. Définissez son [indicateur force-quirks](#) sur *on* . Émettre le jeton actuel. Émettre un jeton de fin de fichier.

##### **Rien d'autre**

Il s'agit d'une [erreur d'analyse missing-whitespace-before-doctype-name](#) . [Reprendre](#) dans l' [état avant le nom DOCTYPE](#) .

#### **13.2.5.54 Avant l'état du nom DOCTYPE**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Ignorez le personnage.

#### **[Alpha supérieur ASCII](#)**

Créez un nouveau jeton DOCTYPE. Définissez le nom du jeton sur la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère). Passez à l' [état du nom DOCTYPE](#) .

**U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Créez un nouveau jeton DOCTYPE. Définissez le nom du jeton sur un caractère U+FFFD REPLACEMENT CHARACTER. Passez à l' [état du nom DOCTYPE](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse de nom de type de document manquant](#) . Créez un nouveau jeton DOCTYPE. Définissez son [indicateur force-quirks](#) sur *on* . Passez à l' [état des données](#) . Émettre le jeton actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . Créez un nouveau jeton DOCTYPE. Définissez son [indicateur force-quirks](#) sur *on* . Émettre le jeton actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**

Créez un nouveau jeton DOCTYPE. Définissez le nom du jeton sur le [caractère d'entrée actuel](#) . Passez à l' [état du nom DOCTYPE](#) .

#### **13.2.5.55 État du nom de DOCTYPE**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

#### **U+000C SAUT DE FORMULAIRE (FF)**

#### **U+0020 ESPACE**

Passez à l' [état après le nom DOCTYPE](#) .

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

#### **Alpha supérieur ASCII**

Ajoutez la version minuscule du [caractère d'entrée actuel](#) (ajoutez 0x0020 au point de code du caractère) au nom du jeton DOCTYPE actuel.

#### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère de remplacement U+FFFD au nom du jeton DOCTYPE actuel.

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Ajoutez le [caractère d'entrée actuel](#) au nom du jeton DOCTYPE actuel.

### **13.2.5.56 Après l'état du nom DOCTYPE**

Consomme le [caractère d'entrée suivant](#) :

#### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

#### **U+000A ALIMENTATION LIGNE (LF)**

#### **U+000C SAUT DE FORMULAIRE (FF)**

#### **U+0020 ESPACE**

Ignorez le personnage.

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Si les six caractères à partir du [caractère d'entrée actuel](#) correspondent à une correspondance [ASCII insensible à la casse](#) pour le mot "PUBLIC", consommez ces caractères et passez à l' [état du mot clé public after DOCTYPE](#) .

Sinon, si les six caractères commençant par le [caractère d'entrée actuel](#) correspondent à une correspondance [ASCII insensible à la casse](#) pour le mot "SYSTEM", utilisez ces caractères et passez à l' [état du mot clé système après DOCTYPE](#) .

Sinon, il s'agit d'une [erreur d'analyse de séquence de caractères non valides après le nom du type de document](#) . [Définissez l' indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . [Reprendre](#) dans le [faux état DOCTYPE](#) .

#### **13.2.5.57 Après l'état du mot clé public DOCTYPE**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Basculer vers l' [état antérieur à l'identifiant public DOCTYPE](#) .

**U+0022 GUILLEMET (")**

Il s'agit d'une [erreur d'analyse missing-whitespace-after-doctype-public-keyword](#) . Définissez l'identifiant public du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant public DOCTYPE \(entre guillemets\)](#) .

**U+0027 APOSTROPHE (')**

Il s'agit d'une [erreur d'analyse missing-whitespace-after-doctype-public-keyword](#) . Définissez l'identifiant public du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant public DOCTYPE \(entre guillemets simples\)](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse missing-doctype-public-identifier](#) . [Définissez l' indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l' indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**

Il s'agit d'une [erreur d'analyse missing-quote-before-doctype-public-identifier](#) . [Définissez l' indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . [Reprendre](#) dans le [faux état DOCTYPE](#) .

### 13.2.5.58 Avant l'état de l'identificateur public DOCTYPE

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Ignorez le personnage.

**U+0022 GUILLEMET (")**

Définissez l'identifiant public du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant public DOCTYPE \(entre guillemets\)](#) .

**U+0027 APOSTROPHE (')**

Définissez l'identifiant public du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant public DOCTYPE \(entre guillemets simples\)](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse missing-doctype-public-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**

Il s'agit d'une [erreur d'analyse missing-quote-before-doctype-public-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . [Reprendre](#) dans le [faux état DOCTYPE](#) .

### 13.2.5.59 État de l'identificateur public DOCTYPE (entre guillemets)

Consomme le [caractère d'entrée suivant](#) :

**U+0022 GUILLEMET (")**

Passez à l' [état d'identifiant public après DOCTYPE](#) .

**U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère U+FFFD REPLACEMENT CHARACTER à l'identifiant public du jeton DOCTYPE actuel.

### **U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse abrupte-doctype-public-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

### **Rien d'autre**

Ajoutez le [caractère d'entrée actuel](#) à l'identifiant public du jeton DOCTYPE actuel.

## **13.2.5.60 État de l'identificateur public DOCTYPE (entre guillemets simples)**

Consomme le [caractère d'entrée suivant](#) :

### **U+0027 APOSTROPHE (')**

Passez à l' [état d'identifiant public après DOCTYPE](#) .

### **U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère U+FFFD REPLACEMENT CHARACTER à l'identifiant public du jeton DOCTYPE actuel.

### **U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse abrupte-doctype-public-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

### **Rien d'autre**

Ajoutez le [caractère d'entrée actuel](#) à l'identifiant public du jeton DOCTYPE actuel.

## **13.2.5.61 Après l'état de l'identificateur public DOCTYPE**

Consomme le [caractère d'entrée suivant](#) :



**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Passez à l' [état entre DOCTYPE public et identifiants système](#) .

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

**U+0022 GUILLEMET (")**

Il s'agit d'une [erreur d'analyse missing-whitespace-between-doctype-public-and-system-identifiers](#) . Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets\)](#) .

**U+0027 APOSTROPHE (')**

Il s'agit d'une [erreur d'analyse missing-whitespace-between-doctype-public-and-system-identifiers](#) . Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets simples\)](#) .

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

**Rien d'autre**

Il s'agit d'une [erreur d'analyse missing-quote-before-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . [Reprendre](#) dans le [faux état DOCTYPE](#) .

### **13.2.5.62 Entre l'état des identifiants public et système de DOCTYPE**

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Ignorez le personnage.

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

**U+0022 GUILLEMET (")**

Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets\)](#) .

#### **U+0027 APOSTROPHE (')**

Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets simples\)](#) .

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### **Rien d'autre**

Il s'agit d'une [erreur d'analyse missing-quote-before-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . [Reprendre](#) dans le [faux état DOCTYPE](#) .

### **13.2.5.63 Après l'état du mot clé système DOCTYPE**

Consomme le [caractère d'entrée suivant](#) :

#### **U+0009 TABULATION DE CARACTÈRES (tabulation)**

#### **U+000A ALIMENTATION LIGNE (LF)**

#### **U+000C SAUT DE FORMULAIRE (FF)**

#### **U+0020 ESPACE**

Passer à l' [état avant l'identifiant système DOCTYPE](#) .

#### **U+0022 GUILLEMET (")**

Il s'agit d'une [erreur d'analyse missing-whitespace-after-doctype-system-keyword](#) . Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets\)](#) .

#### **U+0027 APOSTROPHE (')**

Il s'agit d'une [erreur d'analyse missing-whitespace-after-doctype-system-keyword](#) . Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets simples\)](#) .

#### **U+003E SIGNE SUPÉRIEUR À (>)**

Il s'agit d'une [erreur d'analyse missing-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

#### **EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### Rien d'autre

Il s'agit d'une [erreur d'analyse missing-quote-before-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . [Reprendre](#) dans le [faux état DOCTYPE](#) .

### 13.2.5.64 Avant l'état de l'identificateur de système DOCTYPE

Consomme le [caractère d'entrée suivant](#) :

#### U+0009 TABULATION DE CARACTÈRES (tabulation)

#### U+000A ALIMENTATION LIGNE (LF)

#### U+000C SAUT DE FORMULAIRE (FF)

#### U+0020 ESPACE

Ignorez le personnage.

#### U+0022 GUILLEMET (")

Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets\)](#) .

#### U+0027 APOSTROPHE (')

Définissez l'identifiant système du jeton DOCTYPE actuel sur la chaîne vide (non manquante), puis passez à l' [état de l'identifiant système DOCTYPE \(entre guillemets simples\)](#) .

#### U+003E SIGNE SUPÉRIEUR À (>)

Il s'agit d'une [erreur d'analyse missing-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

#### EOF

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### Rien d'autre

Il s'agit d'une [erreur d'analyse missing-quote-before-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . [Reprendre](#) dans le [faux état DOCTYPE](#) .

### 13.2.5.65 État de l'identificateur de système DOCTYPE (entre guillemets)

Consomme le [caractère d'entrée suivant](#) :

#### U+0022 GUILLEMET (")

Passez à l' [état après l'identifiant système DOCTYPE](#) .

#### U+0000 NUL

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère U+FFFD REPLACEMENT CHARACTER à l'identifiant système du jeton DOCTYPE actuel.

#### U+003E SIGNE SUPÉRIEUR À (>)

Il s'agit d'une [erreur d'analyse abrupte-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

#### EOF

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### Rien d'autre

Ajoutez le [caractère d'entrée actuel](#) à l'identifiant système du jeton DOCTYPE actuel.

### 13.2.5.66 État de l'identificateur système DOCTYPE (entre guillemets simples)

Consomme le [caractère d'entrée suivant](#) :

#### U+0027 APOSTROPHE (')

Passez à l' [état après l'identifiant système DOCTYPE](#) .

#### U+0000 NUL

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ajoutez un caractère U+FFFD REPLACEMENT CHARACTER à l'identifiant système du jeton DOCTYPE actuel.

#### U+003E SIGNE SUPÉRIEUR À (>)

Il s'agit d'une [erreur d'analyse abrupte-doctype-system-identifier](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

#### EOF

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### Rien d'autre

Ajoutez le [caractère d'entrée actuel](#) à l'identifiant système du jeton DOCTYPE actuel.

### 13.2.5.67 Après l'état de l'identificateur de système DOCTYPE

Consomme le [caractère d'entrée suivant](#) :

**U+0009 TABULATION DE CARACTÈRES (tabulation)**

**U+000A ALIMENTATION LIGNE (LF)**

**U+000C SAUT DE FORMULAIRE (FF)**

**U+0020 ESPACE**

Ignorez le personnage.

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton DOCTYPE actuel.

**EOF**

Il s'agit d'une [erreur d'analyse eof-in-doctype](#) . [Définissez l'indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* . Émettez le jeton DOCTYPE actuel. Émettre un jeton de fin de fichier.

#### Rien d'autre

Il s'agit d'une [erreur d'analyse inattendue du caractère après l'identifiant du système de type de document](#) . [Reprendre](#) dans le [faux état DOCTYPE](#) . (Cela ne définit pas l' [indicateur force-quirks](#) du jeton DOCTYPE actuel sur *on* .)

### 13.2.5.68 État DOCTYPE erroné

Consomme le [caractère d'entrée suivant](#) :

**U+003E SIGNE SUPÉRIEUR À (>)**

Passez à l' [état des données](#) . Émettez le jeton DOCTYPE.

**U+0000 NUL**

Il s'agit d'une [erreur d'analyse inattendue de caractères nuls](#) . Ignorez le personnage.

**EOF**

Émettez le jeton DOCTYPE. Émettre un jeton de fin de fichier.

#### Rien d'autre

Ignorez le personnage.

#### 13.2.5.69 État de la section CDATA

Consomme le [caractère d'entrée suivant](#) :

##### U+005D CROCHET DROIT (])

Passez à [l'état des crochets de la section CDATA](#) .

##### EOF

Il s'agit d'une [erreur d'analyse eof-in-cdata](#) . Émettre un jeton de fin de fichier.

##### Rien d'autre

Émet le [caractère d'entrée actuel](#) en tant que jeton de caractère.

*Les caractères U+0000 NULL sont traités dans l'étape de construction de l'arborescence, dans le cadre du mode d'insertion [de contenu étranger](#) , qui est le seul endroit où les sections CDATA peuvent apparaître.*

#### 13.2.5.70 État du support de section CDATA

Consomme le [caractère d'entrée suivant](#) :

##### U+005D CROCHET DROIT (])

Passez à l' [état final de la section CDATA](#) .

##### Rien d'autre

Émettez un jeton de caractère U+005D RIGHT SQUARE BRACKET. [Reprendre](#) dans l' [état de la section CDATA](#) .

#### 13.2.5.71 État final de la section CDATA

Consomme le [caractère d'entrée suivant](#) :

##### U+005D CROCHET DROIT (])

Émettez un jeton de caractère U+005D RIGHT SQUARE BRACKET.

##### U+003E SIGNE SUPÉRIEUR À caractère

Passez à l' [état des données](#) .

##### Rien d'autre

Émettez deux jetons de personnage U+005D RIGHT SQUARE BRACKET. [Reprendre](#) dans l' [état de la section CDATA](#) .

### 13.2.5.72 Etat de référence des caractères

Définissez le [tampon temporaire](#) sur la chaîne vide. Ajoutez un caractère U+0026 AMPERSAND (&) au [tampon temporaire](#) . Consomme le [caractère d'entrée suivant](#) :

#### ASCII alphanumérique

[Reprendre](#) dans l' [état de référence du caractère nommé](#) .

#### **U+0023 NUMÉRO (#)**

Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) . Passez à l' [état de référence de caractère numérique](#) .

#### **Rien d'autre**

[Points de code affleurants consommés comme référence de caractère](#) . [Reprendre](#) dans l' [état de retour](#) .

### 13.2.5.73 Etat de référence du caractère nommé

Consomme le nombre maximal de caractères possible, où les caractères consommés sont l'un des identificateurs de la première colonne de la table des [références de caractères nommés](#) . Ajoutez chaque caractère au [tampon temporaire](#) lorsqu'il est consommé.

#### **S'il y a une correspondance**

Si la référence de caractère a été [consommée dans le cadre d'un attribut](#) et que le dernier caractère correspondant n'est pas un caractère SEMICOLON U+003B (;), et que le [caractère d'entrée suivant](#) est soit un caractère SIGNE ÉGAL U+003D (=) soit un caractère [alphanumérique ASCII](#) , puis, pour des raisons historiques, [vider les points de code consommés comme référence de caractère](#) et passer à l' [état de retour](#) .

Sinon:

1. Si le dernier caractère correspondant n'est pas un caractère U+003B SEMICOLON (;), il s'agit d'une [erreur d'analyse de référence de caractère manquant](#) .
2. Définissez le [tampon temporaire](#) sur la chaîne vide. Ajoutez un ou deux caractères correspondant au nom de référence de caractère (tel qu'il est donné par la deuxième colonne de la table [des références de caractères nommés](#) ) au [tampon temporaire](#) .
3. [Points de code affleurants consommés comme référence de caractère](#) . Passez à l' [état de retour](#) .

**Sinon**

[Points de code affleurants consommés comme référence de caractère](#) . Passez à l' [état d'esperluette ambiguë](#) .

Si le balisage contient (pas dans un attribut) la chaîne `I'm &notit; I tell you`, la référence de caractère est analysée comme "not", comme dans, `I'm -it; I tell you`(et c'est une erreur d'analyse). Mais si le balisage était `I'm &notin; I tell you`, la référence de caractère serait analysée comme "notin;", ce qui entraînerait `I'm ∉ I tell you`(et aucune erreur d'analyse).

Cependant, si le balisage contient la chaîne `I'm &notit; I tell you` dans un attribut, aucune référence de caractère n'est analysée et la chaîne reste intacte (et il n'y a pas d'erreur d'analyse).

#### 13.2.5.74 *Esperluette ambiguë*

Consomme le [caractère d'entrée suivant](#) :

##### [ASCII alphanumérique](#)

Si la référence de caractère a été [consommée dans le cadre d'un attribut](#) , alors ajoutez le [caractère d'entrée actuel](#) à la valeur de l'attribut actuel. Sinon, émettez le [caractère d'entrée actuel](#) en tant que jeton de caractère.

##### **U+003B POINT-virgule (;)**

Il s'agit d'une [erreur d'analyse de référence de caractère de nom inconnu](#) . [Reprendre](#) dans l' [état de retour](#) .

##### **Rien d'autre**

[Reprendre](#) dans l' [état de retour](#) .

#### 13.2.5.75 *Etat de référence des caractères numériques*

Définissez le **code de référence du caractère** sur zéro (0).

Consomme le [caractère d'entrée suivant](#) :

##### **U + 0078 LETTRE MINUSCULE LATINE X**

##### **U + 0058 LETTRE MAJUSCULE LATINE X**

Ajoute le [caractère d'entrée actuel](#) au [tampon temporaire](#) . Passez à l' [état de début de référence de caractère hexadécimal](#) .

##### **Rien d'autre**

[Reprendre](#) dans l' [état de début de référence de caractère décimal](#) .



#### **13.2.5.76 Etat de début de référence de caractère hexadécimal**

Consomme le [caractère d'entrée suivant](#) :

##### **Chiffre hexadécimal ASCII**

[Reprendre](#) dans l' [état de référence des caractères hexadécimaux](#) .

##### **Rien d'autre**

Il s'agit d'une [erreur d'analyse d'absence de chiffres dans la référence de caractère numérique](#) . [Points de code affleurants consommés comme référence de caractère](#) . [Reprendre](#) dans l' [état de retour](#) .

#### **13.2.5.77 Etat de début de référence de caractère décimal**

Consomme le [caractère d'entrée suivant](#) :

##### **Chiffre ASCII**

[Reprendre](#) dans l' [état de référence du caractère décimal](#) .

##### **Rien d'autre**

Il s'agit d'une [erreur d'analyse d'absence de chiffres dans la référence de caractère numérique](#) . [Points de code affleurants consommés comme référence de caractère](#) . [Reprendre](#) dans l' [état de retour](#) .

#### **13.2.5.78 Etat de référence des caractères hexadécimaux**

Consomme le [caractère d'entrée suivant](#) :

##### **Chiffre ASCII**

Multipliez le [code de référence de caractère](#) par 16. Ajoutez une version numérique du [caractère d'entrée actuel](#) (soustrayez 0x0030 du point de code du caractère) au [code de référence de caractère](#) .

##### **Chiffre hexadécimal supérieur ASCII**

Multipliez le [code de référence du caractère](#) par 16. Ajoutez une version numérique du [caractère d'entrée actuel](#) sous forme de chiffre hexadécimal (soustrayez 0x0037 du point de code du caractère) au [code de référence du caractère](#) .

##### **Chiffre hexadécimal inférieur ASCII**

Multipliez le [code de référence du caractère](#) par 16. Ajoutez une version numérique du [caractère d'entrée actuel](#) sous forme de chiffre hexadécimal (soustrayez 0x0057 du point de code du caractère) au [code de référence du caractère](#) .

## U + 003B POINT-virgule

Passez à l' [état final de référence de caractère numérique](#) .

### Rien d'autre

Il s'agit d'une [erreur d'analyse du point-virgule manquant après la référence de caractère](#) . [Reprendre](#) dans l' [état final de référence de caractère numérique](#) .

## 13.2.5.79 Etat de référence du caractère décimal

Consomme le [caractère d'entrée suivant](#) :

### Chiffre ASCII

Multipliez le [code de référence du caractère](#) par 10. Ajoutez une version numérique du [caractère d'entrée actuel](#) (soustrayez 0x0030 du point de code du caractère) au [code de référence du caractère](#) .

## U + 003B POINT-virgule

Passez à l' [état final de référence de caractère numérique](#) .

### Rien d'autre

Il s'agit d'une [erreur d'analyse du point-virgule manquant après la référence de caractère](#) . [Reprendre](#) dans l' [état final de référence de caractère numérique](#) .

## 13.2.5.80 Etat final de référence de caractère numérique

Vérifiez le [code de référence du caractère](#) :

- Si le nombre est 0x00, il s'agit d'une [erreur d'analyse de référence de caractère nul](#) . Définissez le [code de référence du caractère](#) sur 0xFFFD.
- Si le nombre est supérieur à 0x10FFFF, il s'agit d'une [erreur d'analyse de la référence de caractère hors de la plage unicode](#) . Définissez le [code de référence du caractère](#) sur 0xFFFD.
- Si le nombre est un [substitut](#) , il s'agit d'une [erreur d'analyse de référence de caractère de substitution](#) . Définissez le [code de référence du caractère](#) sur 0xFFFD.
- Si le nombre est un [non-caractère](#) , il s'agit d'une [erreur d'analyse de référence de caractère non-caractère](#) .
- Si le nombre est 0x0D ou un [contrôle](#) qui n'est pas [un espace blanc ASCII](#) , il s'agit d'une [erreur d'analyse de référence de caractère de contrôle](#) . Si le numéro est l'un des numéros de la première colonne du tableau suivant, recherchez la ligne avec ce numéro dans la première colonne et définissez

le [code de référence du caractère](#) sur le numéro de la deuxième colonne de cette ligne.

Nombre	Point de code	
0x80	0x20AC	SIGNE EURO (€)
0x82	0x201A	UNIQUE GUILLEMET BAS-9 (,)
0x83	0x0192	LETTRE MINUSCULE LATINE F CROCHET (f)
0x84	0x201E	DOUBLE GUILLEMET BAS-9 (,,)
0x85	0x2026	ELLIPSE HORIZONTALE (...)
0x86	0x2020	DAGUE (†)
0x87	0x2021	DOUBLE POIGNARD (‡)
0x88	0x02C6	LETTRE MODIFICATIVE ACCENT CIRCONFLEXE (ˆ)
0x89	0x2030	SIGNE POUR MILLE (‰)
0x8A	0x0160	LETTRE MAJUSCULE LATINE S CARON (Š)
0x8B	0x2039	GUILLEMET UNIQUE POINTANT VERS LA GAUCHE (‹)
0x8C	0x0152	LIGATURE MAJUSCULE LATINE OE (Œ)
0x8E	0x017D	LETTRE MAJUSCULE LATINE Z CARON (Ž)
0x91	0x2018	GUILLEMET SIMPLE GAUCHE (')
0x92	0x2019	GUILLEMET SIMPLE DROITE (')
0x93	0x201C	GUILLEMET DOUBLE GAUCHE (“)
0x94	0x201D	GUILLEMET DOUBLE DROITE (”)
0x95	0x2022	BALLE (•)
0x96	0x2013	EN TIRET (–)
0x97	0x2014	EM DASH (—)
0x98	0x02DC	PETIT TILDE (˘)
0x99	0x2122	SIGNE DE MARQUE DE COMMERCE (™)
0x9A	0x0161	LETTRE MINUSCULE LATINE S AVEC CARON (š)
0x9B	0x203A	GUILLEMET UNIQUE POINTANT VERS LA DROITE (›)
0x9C	0x0153	LIGATURE MINUSCULE LATINE OE (œ)
0x9E	0x017E	LETTRE MINUSCULE LATINE Z CARON (ž)
0x9F	0x0178	LETTRE MAJUSCULE LATINE Y TRÉMA (ÿ)

Définissez le [tampon temporaire](#) sur la chaîne vide. Ajoutez un point de code égal au [code de référence du caractère](#) au [tampon temporaire](#) . [Points de code affleurants consommés comme référence de caractère](#) . Passez à l' [état de retour](#) .

### 13.2.6 Construction d'arbres

L'entrée de l'étape de construction de l'arborescence est une séquence de jetons provenant de l'étape [de tokenisation](#) . L'étape de construction de l'arbre est associée à un [Document](#) objet DOM lors de la création d'un analyseur. La "sortie" de cette étape consiste à modifier ou à étendre dynamiquement l'arborescence DOM de ce document.

Cette spécification ne définit pas quand un agent utilisateur interactif doit rendre le [Document](#) afin qu'il soit disponible pour l'utilisateur, ou quand il doit commencer à accepter l'entrée de l'utilisateur.

---

Au fur et à mesure que chaque jeton est émis par le tokenizer, l'agent utilisateur doit suivre les étapes appropriées de la liste suivante, connue sous le nom de **répartiteur de construction d'arbre** :

Si la [pile d'éléments ouverts](#) est vide

Si le [nœud actuel ajusté](#) est un élément dans l' [espace de noms HTML](#)

Si le [nœud actuel ajusté](#) est un [point d'intégration de texte MathML](#) et que le jeton est une balise de début dont le nom de balise n'est ni "mglyph" ni "malignmark"

Si le [nœud actuel ajusté](#) est un [point d'intégration de texte MathML](#) et que le jeton est un jeton de caractère

Si le [nœud actuel ajusté](#) est un élément [MathML<sub>annotation-xml</sub>](#) et que le jeton est une balise de début dont le nom de balise est "svg"

Si le [nœud actuel ajusté](#) est un [point d'intégration HTML](#) et le jeton est une balise de début

Si le [nœud actuel ajusté](#) est un [point d'intégration HTML](#) et le jeton est un jeton de caractère

Si le jeton est un jeton de fin de fichier

Traiter le jeton selon les règles données dans la section correspondant au [mode d'insertion](#) courant dans le contenu HTML.

**Sinon**

Traitez le jeton conformément aux règles indiquées dans la section relative à l'analyse des jetons [dans le contenu étranger](#) .

Le **jeton suivant** est le jeton qui est sur le point d'être traité par le [répartiteur de construction d'arbre](#) (même si le jeton est simplement ignoré par la suite).

Un nœud est un **point d'intégration de texte MathML** s'il s'agit de l'un des éléments suivants :

- Un élément [MathML<sub>mi</sub>](#)
- Un élément [MathML<sub>mo</sub>](#)

- Un élément [MathML<sub>mn</sub>](#)
- Un élément [MathML<sub>ms</sub>](#)
- Un élément [MathML<sub>mtext</sub>](#)

Un nœud est un **point d'intégration HTML** s'il s'agit de l'un des éléments suivants :

- Un élément [MathML<sub>annotation-xml</sub>](#) dont le jeton de balise de début avait un attribut avec le nom "encoding" dont la valeur était une correspondance [ASCII insensible à la casse](#) pour la chaîne " text/html "
- Un élément [MathML<sub>annotation-xml</sub>](#) dont le jeton de balise de début avait un attribut avec le nom "encoding" dont la valeur était une correspondance [ASCII insensible à la casse](#) pour la chaîne " application/xhtml+xml "
- Un élément [SVG<sub>foreignObject</sub>](#)
- Un élément [SVG<sub>desc</sub>](#)
- Un élément [SVG<sub>title</sub>](#)

*Si le nœud en question est l'élément [de contexte](#) transmis à l'[algorithme d'analyse de fragment HTML](#), le jeton de balise de début pour cet élément est le "faux" jeton créé par cet [algorithme d'analyse de fragment HTML](#).*

*Tous les noms de balises mentionnés ci-dessous ne sont pas des noms de balises conformes à cette spécification ; beaucoup sont inclus pour gérer le contenu hérité. Ils font toujours partie de l'algorithme que les mises en œuvre doivent mettre en œuvre pour revendiquer la conformité.*

*L'algorithme décrit ci-dessous n'impose aucune limite à la profondeur de l'arborescence DOM générée, ni à la longueur des noms de balises, des noms d'attributs, des valeurs d'attributs, [Text](#) des nœuds, etc. Bien que les implémenteurs soient encouragés à [éviter les limites arbitraires](#), il est reconnu que des problèmes pratiques forcent probablement les agents utilisateurs à imposer des contraintes de profondeur d'imbrication.*

### 13.2.6.1 Création et insertion de nœuds

Pendant que l'analyseur traite un jeton, il peut activer ou désactiver le **parentage adactif**. Cela affecte l'algorithme suivant.

L' **emplacement approprié pour insérer un nœud**, éventuellement à l'aide d'une *cible de remplacement* particulière, est la position dans un élément renvoyé en exécutant les étapes suivantes :

1. Si une *cible de remplacement* a été spécifiée, laissez *target* être la *cible de remplacement*.

Sinon, laissez *target* être le [nœud actuel](#) .

2. Déterminez l' *emplacement d'insertion ajusté* en utilisant les premières étapes correspondantes de la liste suivante :

Si **la famille d'accueil** est activée et que **la cible** est un élément [table](#), [tbody](#), [tfoot](#), [thead](#) ou [tr](#)

*La parentalité adoptive se produit lorsque le contenu est mal imbriqué dans les tableaux.*

Exécutez ces sous-étapes :

1. Soit *last template* le dernier [template](#) élément de la [pile d'éléments ouverts](#) , le cas échéant.
2. Soit *last table* le dernier [table](#) élément de la [pile d'éléments ouverts](#) , le cas échéant.
3. S'il y a un *dernier modèle* et qu'il n'y a pas de *dernière table* , ou qu'il y en a une, mais que le *dernier modèle* est inférieur (ajouté plus récemment) à la *dernière table* dans la [pile d'éléments ouverts](#) , alors : laissez l'*emplacement d'insertion ajusté* être à l'intérieur du *dernier modèle* ' s [template contents](#) , après son dernier enfant (le cas échéant), et abandonnez ces étapes.
4. S'il n'y a pas de *last table* , laissez l'*emplacement d'insertion ajusté* se trouver à l'intérieur du premier élément de la [pile d'éléments ouverts](#) (l' [html](#) élément), après son dernier enfant (le cas échéant), et annulez ces étapes. ( [cas de fragment](#) )
5. Si la *dernière table* a un nœud parent, laissez l'*emplacement d'insertion ajusté* se trouver à l'intérieur du nœud parent de la *dernière table* , juste avant la *dernière table* , et annulez ces étapes.
6. Soit *élément précédent* l'élément immédiatement au-dessus de la *dernière table* dans la [pile des éléments ouverts](#) .
7. Laissez l'*emplacement d'insertion ajusté* être à l'intérieur de l'*élément précédent* , après son dernier enfant (le cas échéant).

*Ces étapes sont impliquées en partie parce qu'il est possible que des éléments, l' [table](#) élément dans ce cas en particulier, aient été déplacés par un script dans le DOM, voire entièrement supprimés du DOM, après l'insertion de l'élément par l'analyseur.*

## Sinon

Laissez l'*emplacement d'insertion ajusté* être à l'intérieur de *target* , après son dernier enfant (le cas échéant).

3. Si l' *emplacement d'insertion ajusté* se trouve à l'intérieur d'un [template](#) élément, laissez-le plutôt à l'intérieur du [contenu du modèle](#) [template](#) de l'élément , après son dernier enfant (le cas échéant).

4. Renvoie l' *emplacement d'insertion ajusté* .

---

Lorsque les étapes ci-dessous nécessitent que l'UA **crée un élément pour un jeton** dans un *espace de noms donné* particulier et avec un *parent prévu* particulier , l'UA doit exécuter les étapes suivantes :

1. Si l' [analyseur HTML spéculatif actif](#) n'est pas null, renvoie le résultat de [la création d'un élément simulé spéculatif avec un namespace donné](#) , le nom de balise du jeton donné et les attributs du jeton donné.
2. Sinon, créez éventuellement [un élément factice spéculatif avec un espace de noms donné](#) , le nom de balise du jeton donné et les attributs du jeton donné.

*Le résultat n'est pas utilisé. Cette étape permet de lancer une [extraction spéculative à partir d'une analyse non spéculative](#). La récupération est encore spéculative à ce stade, car, par exemple, au moment où l'élément est inséré, le parent prévu peut avoir été supprimé du document.*

3. Soit *document* le [nœud document](#) du *parent prévu* .
4. Soit *nom local* le nom de balise du jeton.
5. Soit *is* la valeur de l' [is](#) attribut " " dans le jeton donné, si un tel attribut existe, ou null sinon.
6. Soit *définition* le résultat de [la recherche d'une définition d'élément personnalisée](#) donnée *document* , *espace de noms donné* , *nom local* et *is* .
7. Si *la définition* n'est pas nulle et que l'analyseur n'a pas été créé dans le cadre de l' [algorithme d'analyse de fragment HTML](#) , alors *let will execute script* sera vrai. Sinon, que ce soit faux.
8. Si *exécutera le script* est vrai, alors :
  1. Incrémente [le compteur d'insertion de balisage dynamique](#) du *document* .
  2. Si la [pile de contexte d'exécution JavaScript](#) est vide, [effectuez un point de contrôle de microtâche](#) .
  3. Poussez une nouvelle [file d'attente d'éléments](#) sur [la pile de réactions d'éléments personnalisées](#) de l'[agent concerné](#) du *document* .
9. Soit *element* le résultat de [la création d'un élément](#) donné *document* , *localName* , *namespace donné* , null



et `is` . Si `exécute le script` est vrai, définissez l' *indicateur d'éléments personnalisés synchrones* ; sinon, laissez-le inactif.

*Cela entraînera l'exécution des constructeurs d'éléments personnalisés , si exécute le script est vrai. Cependant, puisque nous avons incrémenté le compteur throw-on-dynamic-markup-insertion , cela ne peut pas entraîner l'insertion de nouveaux caractères dans le tokenizer ou la destruction du document .*

10. Ajoutez chaque attribut du jeton donné à `element` .

*Cela peut mettre en file d'attente une réaction de rappel d'élément personnalisé pour le `attributeChangedCallback`, qui peut s'exécuter immédiatement (à l'étape suivante). Même si l' `is` attribut régit la création d'un élément intégré personnalisé , il n'est pas présent lors de l'exécution du constructeur d'élément personnalisé pertinent ; il est ajouté à cette étape, avec tous les autres attributs.*

11. Si `exécute le script` est vrai, alors :

1. Soit *file d'attente* le résultat de l'extraction de la pile de réactions d'éléments personnalisés de l'agent pertinent du *document* . (Ce sera la même file d'attente d'éléments que celle qui a été poussée ci-dessus.)
2. Invquez des réactions d'élément personnalisées dans la *file d'attente* .
3. Décrémente le compteur d'insertion de balisage dynamique du *document* .

12. Si l'*élément* a un `xmlns`attribut dans l' espace de noms XMLNS dont la valeur n'est pas exactement la même que l'espace de noms de l'élément, c'est une erreur d'analyse . De même, si l'*élément* a un `xmlns:xlink`attribut dans l' espace de noms XMLNS dont la valeur n'est pas l' espace de noms XLink , c'est une erreur d'analyse .

13. Si `element` est un élément réinitialisable , invoquez son algorithme de réinitialisation . (Cela initialise la valeur et la vérification de l'élément en fonction des attributs de l'élément.)

14. Si `element` est un élément associé à un formulaire et non un élément personnalisé associé à un formulaire , le formpointeur d'élément n'est pas nul, il n'y a pas `template`d'élément sur la pile d'éléments ouverts , l'*élément* n'est pas répertorié ou n'a pas d' `form`attribut, et le *parent prévu* est dans le même arbre que l'élément pointé par le formpointeur d'élément , puis associez *élément* à l' `form`élément pointé par le formpointeur d'élément et définissez les éléments de l' élémentindicateur d'insertion de l'analyseur .

15. *Élément* de retour .



---

Lorsque les étapes ci-dessous nécessitent que l'agent utilisateur **insère un élément étranger** pour un jeton dans un espace de noms donné, l'agent utilisateur doit exécuter ces étapes :

1. Laissez l' *emplacement d'insertion ajusté* être l' [endroit approprié pour insérer un nœud](#) .
2. Soit *element* le résultat de [la création d'un élément pour le jeton](#) dans l'espace de noms donné, le parent prévu étant l'élément dans lequel se trouve l' *emplacement d'insertion ajusté* .
3. S'il est possible d'insérer *un élément* à l' *emplacement d'insertion ajusté* , alors :
  1. Si l'analyseur n'a pas été créé dans le cadre de l' [algorithme d'analyse de fragment HTML](#) , placez une nouvelle [file d'attente d'éléments](#) sur [la pile de réactions d'éléments personnalisées](#) de l' [agent concerné de l'élément](#) .
  2. Insérer l'élément à l' *emplacement d'insertion ajusté* .
  3. Si l' analyseur n'a pas été créé dans le cadre de l' [algorithme d' analyse de fragment HTML](#) , puis pop la [file d' éléments](#) de [la pile de réactions d' élément personnalisées](#) de l' [agent concerné de l' élément](#) , et [appelle les réactions d' élément personnalisées](#) dans cette file d' attente.

*Si l' emplacement d'insertion ajusté ne peut pas accepter plus d'éléments, par exemple parce que c'est un [Document](#) qui a déjà un élément enfant, alors l'élément est déposé sur le sol.*

4. Poussez *element* sur la [pile d'éléments ouverts](#) afin qu'il soit le nouveau [nœud courant](#) .
5. *Élément* de retour .

Lorsque les étapes ci-dessous nécessitent que l'agent utilisateur **insère un élément HTML** pour un jeton, l'agent utilisateur doit [insérer un élément étranger](#) pour le jeton, dans l' [espace de noms HTML](#) .

---

Lorsque les étapes ci-dessous nécessitent que l'agent utilisateur **ajuste les attributs MathML** pour un jeton, alors, si le jeton a un attribut nommé `definitionurl`, changez son nom en `definitionURL`(notez la différence de casse).

Lorsque les étapes ci-dessous nécessitent que l'agent utilisateur **ajuste les attributs SVG** pour un jeton, alors, pour chaque attribut sur le jeton dont le nom d'attribut est l'un de ceux de la première colonne du tableau suivant, remplacez le nom de l'attribut par le nom donné dans la cellule correspondante dans la deuxième colonne. (Cela corrige la casse des attributs SVG qui ne sont pas tous en minuscules.)

Nom d'attribut sur le jeton	Nom d'attribut sur l'élément
attributename	attributeName
attributetype	attributeType
basefrequency	baseFrequency
baseprofile	baseProfile
calcmode	calcMode
clippathunits	clipPathUnits
diffuseconstant	diffuseConstant
edgemode	edgeMode
filterunits	filterUnits
glyphref	glyphRef
gradienttransform	gradientTransform
gradientunits	gradientUnits
kernelmatrix	kernelMatrix
kernelunitlength	kernelUnitLength
keypoints	keyPoints
keysplines	keySplines
keytimes	keyTimes
lengthadjust	lengthAdjust
limitingconeangle	limitingConeAngle
markerheight	markerHeight
markerunits	markerUnits
markerwidth	markerWidth
maskcontentunits	maskContentUnits
maskunits	maskUnits
numoctaves	numOctaves
pathlength	pathLength
patterncontentunits	patternContentUnits
patterntransform	patternTransform
patternunits	patternUnits
pointsatx	pointsAtX
pointsaty	pointsAtY
pointsatz	pointsAtZ
preservealpha	preserveAlpha

Nom d'attribut sur le jeton	Nom d'attribut sur l'élément
preserveaspectratio	preserveAspectRatio
primitiveunits	primitiveUnits
refx	refX
refy	refY
repeatcount	repeatCount
repeatdur	repeatDur
requiredextensions	requiredExtensions
requiredfeatures	requiredFeatures
specularconstant	specularConstant
specularexponent	specularExponent
spreadmethod	spreadMethod
startoffset	startOffset
stddeviation	stdDeviation
stitchtiles	stitchTiles
surfacescale	surfaceScale
systemlanguage	systemLanguage
tablevalues	tableValues
targetx	targetX
targety	targetY
textlength	textLength
viewbox	viewBox
viewtarget	viewTarget
xchannelselector	xChannelSelector
ychannelselector	yChannelSelector
zoomandpan	zoomAndPan

Lorsque les étapes ci-dessous nécessitent que l'agent utilisateur ajuste **les attributs étrangers** pour un jeton, alors, si l'un des attributs du jeton correspond aux chaînes données dans la première colonne du tableau suivant, laissez l'attribut être un attribut d'espace de noms, avec le préfixe étant la chaîne donnée dans la cellule correspondante de la deuxième colonne, le nom local étant la chaîne donnée dans la cellule correspondante de la troisième colonne, et l'espace de noms étant l'espace de noms donné dans la cellule correspondante de la quatrième colonne. (Cela corrige l'utilisation des attributs d'espace de noms, en particulier [langles attributs dans l' espace de noms XML](#) .)

Nom d'attribut	Préfixe	Nom local	Espace de noms
xlink:actuate	xlink	actuate	<a href="#">Espace de noms XLink</a>
xlink:arcrole	xlink	arcrole	<a href="#">Espace de noms XLink</a>
xlink:href	xlink	href	<a href="#">Espace de noms XLink</a>
xlink:role	xlink	role	<a href="#">Espace de noms XLink</a>
xlink:show	xlink	show	<a href="#">Espace de noms XLink</a>

Nom d'attribut	Préfixe	Nom local	Espace de noms
xlink:title	xlink	title	<a href="#">Espace de noms XLink</a>
xlink:type	xlink	type	<a href="#">Espace de noms XLink</a>
xml:lang	xml	lang	<a href="#">Espace de noms XML</a>
xml:space	xml	space	<a href="#">Espace de noms XML</a>
xmlns	(aucun)	xmlns	<a href="#">Espace de noms XMLNS</a>
xmlns:xlink	xmlns	xlink	<a href="#">Espace de noms XMLNS</a>

Lorsque les étapes ci-dessous nécessitent que l'agent utilisateur **insère un caractère** lors du traitement d'un jeton, l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit *données* les caractères passés à l'algorithme, ou, si aucun caractère n'a été explicitement spécifié, le caractère du jeton de caractère en cours de traitement.
2. Laissez l' *emplacement d'insertion ajusté* être l' [endroit approprié pour insérer un nœud](#) .
3. Si l' *emplacement d'insertion ajusté* se trouve dans un [Document](#) nœud, alors revenez.

*Le DOM ne laissera pas [Document](#) les nœuds avoir [Text](#) des nœuds enfants, ils sont donc déposés par terre.*

4. S'il existe un [Text](#) nœud juste avant l' *emplacement d'insertion ajusté* , ajoutez des *données* aux [données](#)[Text](#) de ce nœud .

Sinon, créez un nouveau [Text](#) nœud dont [les données](#) sont des *données* et dont [le document de nœud](#) est le même que celui de l'élément dans lequel se trouve l' *emplacement d'insertion ajusté* , et insérez le nœud nouvellement créé à l' *emplacement d'insertion ajusté* .

Voici quelques exemples d'entrées dans l'analyseur et le nombre correspondant de [Text](#) nœuds qui en résultent, en supposant un agent utilisateur qui exécute des scripts.

Saisir	Nombre de <a href="#">Text</a> nœuds
<pre>A&lt;script&gt; var script = document.getElementsByTagName('script') [0];</pre>	Un <a href="#">Text</a> nœud dans le document, contenant "AB".

Saisir	Nombre de <u>Text</u> nœuds
<pre>document.body.removeChild(script); &lt;/script&gt;B</pre>	
<pre>A&lt;script&gt; var text = document.createTextNode('B'); document.body.appendChild(text); &lt;/script&gt;C</pre>	Trois <u>Text</u> nœuds ; "A" avant le script, le contenu du script et "BC" après le script (l'analyseur s'ajoute au <u>Text</u> nœud créé par le script).
<pre>A&lt;script&gt; var text = document.getElementsByTagName('script')[0].firstChild; text.data = 'B'; document.body.appendChild(text); &lt;/script&gt;C</pre>	Deux nœuds adjacents <u>Text</u> dans le document, contenant "A" et "BC".
<pre>A&lt;table&gt;B&lt;tr&gt;C&lt;/tr&gt;D&lt;/table&gt;</pre>	Un <u>Text</u> nœud avant la table, contenant "ABCD". (Ceci est causé par <a href="#">la famille d'accueil</a> .)
<pre>A&lt;table&gt;&lt;tr&gt; B&lt;/tr&gt; C&lt;/table&gt;</pre>	Un <u>Text</u> nœud avant la table, contenant "A B C" (A-espace-B-espace-C). (Ceci est causé par <a href="#">la famille d'accueil</a> .)
<pre>A&lt;table&gt;&lt;tr&gt; B&lt;/tr&gt; &lt;/em&gt;C&lt;/table&gt;</pre>	Un <u>Text</u> nœud avant la table, contenant "A BC" (A-space-BC), et un <u>Text</u> nœud à l'intérieur de la table (en tant qu'enfant de a <u>tbody</u> ) avec un seul espace. (Les caractères d'espace séparés des caractères

Saisir	Nombre de <u>Text</u> nœuds
	non-espace par des jetons non-caractères ne sont pas affectés par <a href="#">le parentage adoptif</a> , même si ces autres jetons sont ensuite ignorés.)

Lorsque les étapes ci-dessous nécessitent que l'agent utilisateur **insère un commentaire** lors du traitement d'un jeton de commentaire, éventuellement avec une *position* d'insertion explicite position , l'agent utilisateur doit exécuter les étapes suivantes :

1. Soit *données* les données fournies dans le jeton de commentaire en cours de traitement.
2. Si *position* a été spécifié, laissez l' *emplacement d'insertion ajusté* être *position* . Sinon, laissez l'*emplacement d'insertion ajusté* être l' [endroit approprié pour insérer un nœud](#) .
3. Créez un Commentnœud dont *data*l'attribut est défini sur *data* et dont [le document de nœud](#) est le même que celui du nœud dans lequel se trouve l' *emplacement d'insertion ajusté* .
4. Insérez le nœud nouvellement créé à l' *emplacement d'insertion ajusté* .

Les événements de mutation DOM ne doivent pas se déclencher pour les changements causés par l'UA analysant le document. Cela inclut l'analyse de tout contenu inséré à l'aide des appels `document.write()` et `document.writeln()`. [\[UIÉVÉNEMENTS\]](#)

Cependant, [les observateurs de mutation](#) se déclenchent, comme l'exige *DOM* .

### 13.2.6.2 Analyser des éléments qui ne contiennent que du texte

L' **algorithme générique d'analyse d'éléments de texte brut** et l' **algorithme générique d'analyse d'éléments RCDATA** consistent en les étapes suivantes. Ces algorithmes sont toujours appelés en réponse à un jeton de balise de début.

1. [Insérez un élément HTML](#) pour le jeton.
2. Si l'algorithme invoqué est l' [algorithme générique d'analyse d'éléments de texte brut](#) , basculez le tokenizer à l' [état RAWTEXT](#) ; sinon l'algorithme invoqué était l' [algorithme générique d'analyse d'éléments RCDATA](#) , passez le tokenizer à l' [état RCDATA](#) .
3. Soit le [mode d'insertion d'origine](#) le [mode d'insertion](#) courant .
4. Ensuite, basculez le [mode d'insertion](#) sur « [texte](#) ».

### 13.2.6.3 Éléments de fermeture qui ont des balises de fin implicites

Lorsque les étapes ci-dessous nécessitent que l'UA **génère des balises de fin implicites** , alors, alors que le [nœud actuel](#) est un [dd](#)élément, un [dt](#)élément, un [li](#)élément, un [optgroup](#)élément, un [option](#)élément, un [p](#)élément, un [rb](#)élément, un [element](#) ou un [element](#) , l'UA doit retirer le [nœud courant](#) de la [pile d'éléments ouverts](#) .[rprtrtc](#)

Si une étape nécessite que l'UA génère des balises de fin implicites mais énumère un élément à exclure du processus, alors l'UA doit exécuter les étapes ci-dessus comme si cet élément ne figurait pas dans la liste ci-dessus.

Lorsque les étapes ci-dessous exigent que l'UA **génère soigneusement toutes les balises de fin implicites** , alors, alors que le [nœud actuel](#) est un [caption](#)élément, un [colgroup](#)élément, un [dd](#)élément, un [dt](#)élément, un [li](#)élément, un [optgroup](#)élément, un [option](#)élément, un [p](#)élément, un [rb](#)élément, un [rp](#)élément, un [rt](#) élément, un [rtc](#)élément, [tbody](#)un élément [td](#), un élément [tfoot](#), un élément, un élément, [th](#)un [thead](#)élément ou un [tr](#)élément, l'UA doit retirer le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

### 13.2.6.4 Les règles d'analyse des jetons dans le contenu HTML

#### 13.2.6.4.1 Le mode d'insertion " initiale "

Un [Document](#) objet a un analyseur associé **ne peut pas changer l'indicateur de mode** (un booléen). C'est d'abord faux.

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " initial "](#), l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

Ignorez le jeton.

**Un jeton de commentaire**

[Insérez un commentaire](#) comme dernier enfant de l' [Document](#) objet.

**Un jeton DOCTYPE**

Si le nom du jeton DOCTYPE n'est pas " `html`", ou si l'identifiant public du jeton n'est pas manquant, ou si l'identifiant système du jeton n'est ni manquant ni " `about:legacy-compat`", alors il y a une [erreur d'analyse](#) .

Ajouter un [DocumentType](#) nœud au [Document](#) nœud, avec son [nom](#) défini sur le nom donné dans le jeton DOCTYPE, ou la chaîne vide si le nom était manquant ; son [identifiant public](#) défini sur l'identifiant public donné dans le jeton DOCTYPE, ou la chaîne vide si l'identifiant public était manquant ; et son [ID système](#) défini sur l'identifiant système donné dans le jeton DOCTYPE, ou la chaîne vide si l'identifiant système était manquant.

*Cela garantit également que le [DocumentType](#) nœud est renvoyé en tant que valeur de l' [doctype](#) attribut de l' [Document](#) objet.*

Ensuite, si le document n'est pas [un iframe srcdoc document](#), et que l' [analyseur ne peut pas changer l'indicateur de mode](#) est faux, et que le jeton DOCTYPE correspond à l'une des conditions de la liste suivante, définissez alors le [mode](#)[Document](#) bizarreries :

- Le [drapeau force-quirks](#) est défini sur `on` .
- Le nom n'est pas " `html`".
- L'identifiant public est défini sur : "  `-//W3O//DTD W3 HTML Strict 3.0//EN//`"
- L'identifiant public est défini sur : "  `-/W3C/DTD HTML 4.0 Transitional/EN`"
- L'identifiant public est défini sur : "  `HTML`"
- L'identifiant système est défini sur : "  `http://www.ibm.com/data/dtd/v11/ibmxhtml1-transitional.dtd`"
- L'identifiant public commence par : "  `+//Silmaril//dtd html Pro v0r11 19970101//`"
- L'identifiant public commence par : "  `-//AS//DTD HTML 3.0 asWedit + extensions//`"
- L'identifiant public commence par : "  `-//AdvaSoft Ltd//DTD HTML 3.0 asWedit + extensions//`"
- L'identifiant public commence par : "  `-//IETF//DTD HTML 2.0 Level 1//`"



- L'identifiant public commence par : "`--//IETF//DTD HTML 2.0 Level 2//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 2.0 Strict Level 1//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 2.0 Strict Level 2//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 2.0 Strict//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 2.0//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 2.1E//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 3.0//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 3.2 Final//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 3.2//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML 3//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Level 0//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Level 1//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Level 2//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Level 3//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Strict Level 0//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Strict Level 1//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Strict Level 2//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Strict Level 3//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML Strict//`"
- L'identifiant public commence par : "`--//IETF//DTD HTML//`"
- L'identifiant public commence par : "`--//Metrius//DTD Metrius Presentational//`"
- L'identifiant public commence par : "`--//Microsoft//DTD Internet Explorer 2.0 HTML Strict//`"
- L'identifiant public commence par : "`--//Microsoft//DTD Internet Explorer 2.0 HTML//`"
- L'identifiant public commence par : "`--//Microsoft//DTD Internet Explorer 2.0 Tables//`"
- L'identifiant public commence par : "`--//Microsoft//DTD Internet Explorer 3.0 HTML Strict//`"
- L'identifiant public commence par : "`--//Microsoft//DTD Internet Explorer 3.0 HTML//`"
- L'identifiant public commence par : "`--//Microsoft//DTD Internet Explorer 3.0 Tables//`"
- L'identifiant public commence par : "`--//Netscape Comm. Corp.//DTD HTML//`"
- L'identifiant public commence par : "`--//Netscape Comm. Corp.//DTD Strict HTML//`"
- L'identifiant public commence par : "`--//O'Reilly and Associates//DTD HTML 2.0//`"
- L'identifiant public commence par : "`--//O'Reilly and Associates//DTD HTML Extended 1.0//`"

- L'identifiant public commence par : "`--//O'Reilly and Associates//DTD HTML Extended Relaxed 1.0//`"
- L'identifiant public commence par : "`--//SQ//DTD HTML 2.0 HoTMetaL + extensions//`"
- L'identifiant public commence par : "`--//SoftQuad Software//DTD HoTMetaL PRO 6.0::19990601::extensions to HTML 4.0//`"
- L'identifiant public commence par : "`--//SoftQuad//DTD HoTMetaL PRO 4.0::19971010::extensions to HTML 4.0//`"
- L'identifiant public commence par : "`--//Spyglass//DTD HTML 2.0 Extended//`"
- L'identifiant public commence par : "`--//Sun Microsystems Corp.//DTD HotJava HTML//`"
- L'identifiant public commence par : "`--//Sun Microsystems Corp.//DTD HotJava Strict HTML//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML 3 1995-03-24//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML 3.2 Draft//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML 3.2 Final//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML 3.2//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML 3.2S Draft//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML 4.0 Frameset//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML 4.0 Transitional//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML Experimental 19960712//`"
- L'identifiant public commence par : "`--//W3C//DTD HTML Experimental 970421//`"
- L'identifiant public commence par : "`--//W3C//DTD W3 HTML//`"
- L'identifiant public commence par : "`--//W3O//DTD W3 HTML 3.0//`"
- L'identifiant public commence par : "`--//WebTechs//DTD Mozilla HTML 2.0//`"
- L'identifiant public commence par : "`--//WebTechs//DTD Mozilla HTML//`"
- L'identifiant système est manquant et l'identifiant public commence par : "`--//W3C//DTD HTML 4.01 Frameset//`"
- L'identifiant système est manquant et l'identifiant public commence par : "`--//W3C//DTD HTML 4.01 Transitional//`"

Sinon, si le document n'est pas [un `iframe srcdoc`document](#), et que l'[analyseur ne peut pas changer l'indicateur de mode](#) est faux, et que le jeton DOCTYPE correspond à l'une des conditions de la liste suivante, alors réglez le [Document](#) sur [mode limited-quirks](#) :

- L'identifiant public commence par : "`--//W3C//DTD XHTML 1.0 Frameset//`"
- L'identifiant public commence par : "`--//W3C//DTD XHTML 1.0 Transitional//`"
- L'identifiant système n'est pas manquant et l'identifiant public commence par : "`--//W3C//DTD HTML 4.01 Frameset//`"

- L'identifiant système n'est pas manquant et l'identifiant public commence par : " `-//W3C//DTD HTML 4.01 Transitional//`"

Les chaînes d'identifiant système et d'identifiant public doivent être comparées aux valeurs données dans les listes ci-dessus sans [tenir compte de la casse ASCII](#) . Un identifiant système dont la valeur est la chaîne vide n'est pas considéré comme manquant aux fins des conditions ci-dessus.

Ensuite, basculez le [mode d'insertion](#) sur "[avant html](#)" .

### Rien d'autre

Si le document *n'est pas* [un iframe srcdocdocument](#) , il s'agit d'une [erreur d'analyse](#) ; si l' [analyseur ne peut pas changer l'indicateur de mode](#) est faux, définissez le mode [Document](#) sur [quirks](#) .

Dans tous les cas, passez le [mode d'insertion](#) en « [avant html](#) », puis retraitez le jeton.

#### 13.2.6.4.2 Le mode d'insertion "[avant html](#)"

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " avant html "](#) , l'agent utilisateur doit gérer le jeton comme suit :

#### Un jeton DOCTYPE

[Erreur d'analyse](#) . Ignorez le jeton.

#### Un jeton de commentaire

[Insérez un commentaire](#) comme dernier enfant de l' [Document](#) objet.

#### Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE

Ignorez le jeton.

#### Une balise de début dont le nom de balise est "html"

[Créez un élément pour le jeton](#) dans l' [espace de noms HTML](#) , avec [Document](#) comme parent prévu. Ajoutez-le à l' [Document](#) objet. Placez cet élément dans la [pile des éléments ouverts](#) .

Basculez le [mode d'insertion](#) sur "[avant la tête](#)" .

#### Une balise de fin dont le nom de balise est l'un des suivants : "head", "body", "html", "br"

Agissez comme décrit dans l'entrée "autre chose" ci-dessous.

#### Toute autre balise de fin

[Erreur d'analyse](#) . Ignorez le jeton.

#### Rien d'autre

Créez un [html](#) élément dont [le nœud document](#) est l' [Document](#) objet. Ajoutez-le à l' [Document](#) objet. Placez cet élément dans la [pile des éléments ouverts](#) .

Basculez le [mode d'insertion](#) sur " [avant tête](#) ", puis retraitez le jeton.

L' [élément document](#) peut finir par être supprimé de l' [Document](#) objet, par exemple par des scripts ; rien de particulier ne se produit dans de tels cas, le contenu continue d'être ajouté aux nœuds comme décrit dans la section suivante.

#### 13.2.6.4.3 Le mode d'insertion " avant tête "

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " avant tête "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

Ignorez le jeton.

**Un jeton de commentaire**

[Insérez un commentaire](#) .

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Une balise de début dont le nom de balise est "head"**

[Insérez un élément HTML](#) pour le jeton.

Définissez le [headpointeur d'élément](#) sur l' [head](#) élément nouvellement créé.

Basculez le [mode d'insertion](#) sur " [in head](#) " .

**Une balise de fin dont le nom de balise est l'un des suivants : "head", "body", "html", "br"**

Agissez comme décrit dans l'entrée "autre chose" ci-dessous.

**Toute autre balise de fin**

[Erreur d'analyse](#) . Ignorez le jeton.

## Rien d'autre

[Insérez un élément HTML](#) pour un jeton de balise de début "head" sans attributs.

Définissez le [headpointeur d'élément](#) sur l' [head](#)élément nouvellement créé.

Basculez le [mode d'insertion](#) sur " [in head](#) ".

Retraitez le jeton actuel.

### 13.2.6.4.4 Le mode d'insertion " en tête "

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " in head "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

[Insérez le caractère](#) .

**Un jeton de commentaire**

[Insérez un commentaire](#) .

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Une balise de début dont le nom de balise est l'un des suivants : "base", "basefont", "bgsound", "link"**

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaitre l' indicateur de fermeture automatique du jeton](#) , s'il est défini.

**Une balise de début dont le nom de balise est "meta"**

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaitre l' indicateur de fermeture automatique du jeton](#) , s'il est défini.

Si l' [analyseur HTML spéculatif actif](#) est nul, alors :

1. Si l'élément a un `charset` attribut et [que l'obtention d'un encodage](#) à partir de sa valeur donne un `encoding` et que la `confiance` est actuellement *provisoire* , [remplacez l'encodage](#) par l'encodage résultant.
2. Sinon, si l'élément a un `http-equiv` attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " `Content-Type`", et que l'élément a un `content` attribut, et que l'application de l' [algorithme d'extraction d'un codage de caractères d'un meta-élément](#) à la valeur de cet attribut renvoie un `encoding` , et la `confiance` est actuellement *provisoire* , puis [remplacez le codage](#) par le codage extrait.

*L' [analyseur HTML spéculatif](#) n'applique pas de manière spéculative les déclarations d'encodage de caractères afin de réduire la complexité de l'implémentation.*

**Une balise de début dont le nom de balise est "title"**

Suivez l' [algorithme générique d'analyse d'éléments RCDATA](#) .

**Une balise de début dont le nom de balise est "noscript", si l' [indicateur de script](#) est activé**

**Une balise de début dont le nom de balise est l'un des suivants : "noframes", "style"**

Suivez l' [algorithme générique d'analyse d'éléments de texte brut](#) .

**Une balise de début dont le nom de balise est "noscript", si l' [indicateur de script](#) est désactivé**

[Insérez un élément HTML](#) pour le jeton.

Basculez le [mode d'insertion](#) sur " [in head noscript](#) ".

**Une balise de début dont le nom de balise est "script"**

Exécutez ces étapes :

1. Laissez l' *emplacement d'insertion ajusté* être l' [endroit approprié pour insérer un nœud](#) .
2. [Créez un élément pour le jeton](#) dans l' [espace de noms HTML](#) , le parent prévu étant l'élément dans lequel se trouve l' *emplacement d'insertion ajusté* .
3. Définissez le [document de l'analyseur](#) de l'élément sur `Document`, et définissez la [force asynchrone](#) de l'élément sur `false`.

*Cela garantit que, si le script est externe, tous `document.write()` les appels dans le script s'exécuteront en ligne, au lieu de détruire le document, comme cela se produirait dans la plupart des autres*

*cas. Cela empêche également le script de s'exécuter tant que la balise de fin n'est pas visible.*

4. Si l'analyseur a été créé dans le cadre de l' [algorithme d'analyse de fragment HTML](#) , définissez l' [script](#) élément [déjà commencé](#) sur true. ( [cas de fragment](#) )
5. Si l'analyseur a été appelé via les méthodes `document.write()` ou `document.writeln()` , définissez éventuellement l' [script](#) élément [déjà commencé](#) sur true. (Par exemple, l'agent utilisateur peut utiliser cette clause pour empêcher l'exécution de scripts [cross-origin](#) insérés via `document.write()` dans des conditions de réseau lent, ou lorsque la page a déjà mis longtemps à se charger.)
6. Insérez l'élément nouvellement créé à l' *emplacement d'insertion ajusté* .
7. Poussez l'élément sur la [pile d'éléments ouverts](#) afin qu'il soit le nouveau [nœud courant](#) .
8. Basculez le tokenizer vers l' [état des données de script](#) .
9. Soit le [mode d'insertion d'origine](#) le [mode d'insertion](#) courant .
10. Basculez le [mode d'insertion](#) sur « [texte](#) ».

#### **Une balise de fin dont le nom de balise est "head"**

Détachez le [nœud actuel](#) (qui sera l' [head](#) élément) de la [pile d'éléments ouverts](#) .

Basculez le [mode d'insertion](#) sur " [après la tête](#) ".

#### **Une balise de fin dont le nom de balise est l'un des suivants : "body", "html", "br"**

Agissez comme décrit dans l'entrée "autre chose" ci-dessous.

#### **Une balise de début dont le nom de balise est "template"**

[Insérez un élément HTML](#) pour le jeton.

Insérer un [marqueur](#) à la fin de la [liste des éléments de formatage actifs](#) .

Définissez le [drapeau frameset-ok](#) sur "pas ok".

Basculez le [mode d'insertion](#) sur " [dans le modèle](#) ".

Poussez « [dans le modèle](#) » sur la [pile des modes d'insertion de modèle](#) afin qu'il soit le nouveau [mode d'insertion de modèle actuel](#) .

#### **Une balise de fin dont le nom de balise est "template"**

S'il n'y a pas template d'élément sur la pile d'éléments ouverts , alors c'est une erreur d'analyse ; ignorer le jeton.

Sinon, exécutez ces étapes :

1. Générez soigneusement toutes les balises de fin implicites .
2. Si le nœud actuel n'est pas un template élément, il s'agit d'une erreur d'analyse .
3. Extraire des éléments de la pile d'éléments ouverts jusqu'à ce qu'un template élément ait été extrait de la pile.
4. Efface la liste des éléments de formatage actifs jusqu'au dernier marqueur .
5. Retirez le mode d'insertion de modèle actuel de la pile des modes d'insertion de modèle .
6. Réinitialisez le mode d'insertion de manière appropriée .

**Une balise de début dont le nom de balise est "head"**

**Toute autre balise de fin**

Erreur d'analyse . Ignorez le jeton.

**Rien d'autre**

Détachez le nœud actuel (qui sera l' head élément) de la pile d'éléments ouverts .

Basculez le mode d'insertion sur " après la tête " .

Retraitez le jeton.

#### ***13.2.6.4.5 Le mode d'insertion " in head noscript "***

Lorsque l'agent utilisateur doit appliquer les règles du mode d'insertion " in head noscript " , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton DOCTYPE**

Erreur d'analyse . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton en utilisant les règles du mode d'insertion « dans le corps » .

**Une balise de fin dont le nom de balise est "noscript"**



Pop le [nœud courant](#) (qui sera un [noscript](#)élément) de la [pile des éléments ouverts](#) ; le nouveau [nœud courant](#) sera un [head](#)élément.

Basculez le [mode d'insertion](#) sur " [in head](#) ".

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

**Un jeton de commentaire**

**Une balise de début dont le nom de balise est l'un des suivants : "basefont", "bgsound", "link", "meta", "noframes", "style"**

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [in head](#) " .

**Une balise de fin dont le nom de balise est "br"**

Agissez comme décrit dans l'entrée "autre chose" ci-dessous.

**Une balise de début dont le nom de balise est l'un des suivants : "head", "noscript"**

**Toute autre balise de fin**

[Erreur d'analyse](#) . Ignorez le jeton.

**Rien d'autre**

[Erreur d'analyse](#) .

Pop le [nœud courant](#) (qui sera un [noscript](#)élément) de la [pile des éléments ouverts](#) ; le nouveau [nœud courant](#) sera un [head](#)élément.

Basculez le [mode d'insertion](#) sur " [in head](#) ".

Retraitez le jeton.

#### ***13.2.6.4.6 Le mode d'insertion " après tête "***

Lorsque l'agent utilisateur doit appliquer les règles pour le [mode d'insertion « after head »](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

[Insérez le caractère](#) .

**Un jeton de commentaire**

[Insérez un commentaire](#) .

## Un jeton DOCTYPE

[Erreur d'analyse](#) . Ignorez le jeton.

## Une balise de début dont le nom de balise est "html"

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

## Une balise de début dont le nom de balise est "body"

[Insérez un élément HTML](#) pour le jeton.

Définissez le [drapeau frameset-ok](#) sur "pas ok".

Basculez le [mode d'insertion](#) sur " [dans le corps](#) " .

## Une balise de début dont le nom de balise est "frameset"

[Insérez un élément HTML](#) pour le jeton.

Basculez le [mode d'insertion](#) sur " [in frameset](#) " .

## Une balise de début dont le nom de balise est l'un des suivants : "base", "basefont", "bgsound", "link", "meta", "noframes", "script", "style", "template", "title"

[Erreur d'analyse](#) .

Poussez le nœud pointé par le [headpointeur d'élément](#) sur la [pile d'éléments ouverts](#) .

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [in head](#) " .

Supprimez le nœud pointé par le [headpointeur d'élément](#) de la [pile d'éléments ouverts](#) . (Ce n'est peut-être pas le [nœud actuel](#) à ce stade.)

*Le [headpointeur d'élément](#) ne peut pas être nul à ce stade.*

## Une balise de fin dont le nom de balise est "template"

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [in head](#) " .

## Une balise de fin dont le nom de balise est l'un des suivants : "body", "html", "br"

Agissez comme décrit dans l'entrée "autre chose" ci-dessous.

## Une balise de début dont le nom de balise est "head"

### Toute autre balise de fin

[Erreur d'analyse](#) . Ignorez le jeton.

## Rien d'autre

[Insérez un élément HTML](#) pour un jeton de balise de début "body" sans attribut.

Basculez le [mode d'insertion](#) sur " [dans le corps](#) ".

Retraitez le jeton actuel.

#### **13.2.6.4.7 Le mode d'insertion " dans le corps "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion « in body »](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère qui est U+0000 NULL**

[Erreur d'analyse](#) . Ignorez le jeton.

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez le caractère du jeton](#) .

**Tout autre jeton de personnage**

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez le caractère du jeton](#) .

Définissez le [drapeau frameset-ok](#) sur "pas ok".

**Un jeton de commentaire**

[Insérez un commentaire](#) .

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

[Erreur d'analyse](#) .

S'il y a un [template](#) élément sur la [pile d'éléments ouverts](#) , ignorez le jeton.

Sinon, pour chaque attribut sur le jeton, vérifiez si l'attribut est déjà présent sur l'élément supérieur de la [pile d'éléments ouverts](#) . Si ce n'est pas le cas, ajoutez l'attribut et sa valeur correspondante à cet élément.

**Une balise de début dont le nom de balise est l'un des suivants : "base", "basefont", "bgsound", "link", "meta", "noframes", "script", "style", "template", "title"**

**Une balise de fin dont le nom de balise est "template"**

Traiter le jeton [en suivant les règles du mode d'insertion " in head "](#) .

### Une balise de début dont le nom de balise est "body"

[Erreur d'analyse](#) .

Si le deuxième élément de la [pile d'éléments ouverts](#) n'est pas un [body](#) élément, si la [pile d'éléments ouverts](#) n'a qu'un seul nœud, ou s'il y a un [template](#) élément sur la [pile d'éléments ouverts](#) , alors ignorez le jeton. ( [cas de fragment](#) )

Sinon, définissez l' [indicateur frameset-ok](#) sur "not ok" ; puis, pour chaque attribut sur le jeton, vérifiez si l'attribut est déjà présent sur l' [body](#) élément (le deuxième élément) sur la [pile d'éléments ouverts](#) , et si ce n'est pas le cas, ajoutez l'attribut et sa valeur correspondante à cet élément.

### Une balise de début dont le nom de balise est "frameset"

[Erreur d'analyse](#) .

Si la [pile d'éléments ouverts](#) ne comporte qu'un seul nœud, ou si le deuxième élément de la [pile d'éléments ouverts](#) n'est pas un [body](#) élément, alors ignorez le jeton. ( [cas de fragment](#) )

Si l' [indicateur frameset-ok](#) est défini sur "not ok", ignorez le jeton.

Sinon, exécutez les étapes suivantes :

1. Supprimez le deuxième élément de la [pile d'éléments ouverts](#) de son nœud parent, s'il en a un.
2. Faites éclater tous les nœuds du bas de la [pile d'éléments ouverts](#) , du [nœud actuel](#) jusqu'à l' [html](#) élément racine, mais sans l'inclure.
3. [Insérez un élément HTML](#) pour le jeton.
4. Basculez le [mode d'insertion](#) sur " [in frameset](#) " .

### Un jeton de fin de fichier

Si la [pile des modes d'insertion de modèle](#) n'est pas vide, traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le modèle](#) » .

Sinon, suivez ces étapes :

1. S'il y a un nœud dans la [pile d'éléments ouverts](#) qui n'est ni un [dd](#) élément, un [dt](#) élément, un [li](#) élément, un [optgroup](#) élément, un élément, un [option](#) élément, un [p](#) élément, un élément [rb](#), un [rp](#) élément, un [rt](#) élément [rtc](#), un [tbody](#) élément, un [td](#) élément , un [tfoot](#) élément, un [th](#) élément, un [thead](#) élément, un [tr](#) élément, l' [body](#) élément ou l' [html](#) élément, alors c'est une [erreur d'analyse](#) .

## 2. [Arrêtez l'analyse](#) .

### Une balise de fin dont le nom de balise est "body"

Si la [pile d'éléments ouverts](#) n'a pas [d' bodyélément dans la portée](#) , c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, s'il y a un nœud dans la [pile d'éléments ouverts](#) qui n'est ni un [dd](#)élément, un [dt](#)élément, un [li](#)élément, un [optgroup](#)élément, un élément, un [option](#)élément, un [p](#)élément, un [rb](#)élément, un [rp](#)élément [rt](#), un [rtc](#)élément, un [tbody](#)élément, un [td](#)élément, un [tfoot](#)élément, un [th](#)élément, un [thead](#)élément, un [tr](#)élément, l' [body](#)élément ou l' [html](#)élément, alors c'est une [erreur d'analyse](#) .

Basculez le [mode d'insertion](#) sur " [après le corps](#) " .

### Une balise de fin dont le nom de balise est "html"

Si la [pile d'éléments ouverts](#) n'a pas [d' bodyélément dans la portée](#) , c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, s'il y a un nœud dans la [pile d'éléments ouverts](#) qui n'est ni un [dd](#)élément, un [dt](#)élément, un [li](#)élément, un [optgroup](#)élément, un élément, un [option](#)élément, un [p](#)élément, un [rb](#)élément, un [rp](#)élément [rt](#), un [rtc](#)élément, un [tbody](#)élément, un [td](#)élément, un [tfoot](#)élément, un [th](#)élément, un [thead](#)élément, un [tr](#)élément, l' [body](#)élément ou l' [html](#)élément, alors c'est une [erreur d'analyse](#) .

Basculez le [mode d'insertion](#) sur " [après le corps](#) " .

Retraitez le jeton.

### Une balise de début dont le nom de balise est l'un des suivants : "address", "article", "aside", "blockquote", "center", "details", "dialog", "dir", "div", "dl", "fieldset", "figcaption", "figure", "footer", "header", "hgroup", "main", "menu", "nav", "ol", "p", "section", "summary", "ul"

Si la [pile d'éléments ouverts a un pélément dans la portée du bouton](#) , alors [fermez un p élément](#) .

[Insérez un élément HTML](#) pour le jeton.

### Une balise de début dont le nom de balise est l'un des suivants : "h1", "h2", "h3", "h4", "h5", "h6"

Si la [pile d'éléments ouverts a un pélément dans la portée du bouton](#) , alors [fermez un p élément](#) .

Si le [nœud actuel](#) est un [élément HTML](#) dont le nom de balise est l'un des "h1", "h2", "h3", "h4", "h5" ou "h6", alors il s'agit d'une erreur d' [analyse](#) ; pop le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Insérez un élément HTML](#) pour le jeton.

**Une balise de début dont le nom de balise est l'un des suivants : "pre", "listing"**

Si la [pile d'éléments ouverts a un p élément dans la portée du bouton](#) ,  
alors [fermez un p élément](#) .

[Insérez un élément HTML](#) pour le jeton.

Si le [jeton suivant](#) est un jeton de caractère U+000A LINE FEED (LF), alors ignorez ce jeton et passez au suivant. (Les retours à la ligne au début des [pre](#) blocs sont ignorés pour des raisons de commodité de création.)

Définissez le [drapeau frameset-ok](#) sur "pas ok".

**Une balise de début dont le nom de balise est "form"**

Si le [form](#)[pointeur d'élément](#) n'est pas nul et qu'il n'y a pas [template](#) d'élément sur la [pile d'éléments ouverts](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon:

Si la [pile d'éléments ouverts a un p élément dans la portée du bouton](#) ,  
alors [fermez un p élément](#) .

[Insérez un élément HTML](#) pour le jeton et, s'il n'y a pas [template](#) d'élément sur la [pile d'éléments ouverts](#) , définissez le [form](#)[pointeur d'élément](#) pour qu'il pointe vers l'élément créé.

**Une balise de début dont le nom de balise est "li"**

Exécutez ces étapes :

1. Définissez le [drapeau frameset-ok](#) sur "pas ok".
2. Initialiser *le nœud* pour qu'il soit le [nœud actuel](#) (le nœud le plus bas de la pile).
3. *Boucle* : si *le nœud* est un [li](#) élément, exécutez ces sous-étapes :
  1. [Générez des balises de fin implicites](#) , sauf pour [li](#) les éléments.
  2. Si le [nœud actuel](#) n'est pas un [li](#) élément, il s'agit d'une [erreur d'analyse](#) .
  3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [li](#) élément ait été extrait de la pile.
  4. Passez à l'étape indiquée *ci* -dessous.
4. Si *le nœud* est dans la catégorie [spéciale](#) , mais n'est pas un élément [address](#), [div](#), ou [p](#), passez à l'étape intitulée *done* ci-dessous.

5. Sinon, définissez *node* sur l'entrée précédente dans la [pile d'éléments ouverts](#) et revenez à l'étape étiquetée *loop* .
6. *Terminé* : si la [pile d'éléments ouverts contient un pélément dans la portée du bouton](#) , [fermez un pélément](#) .
7. Enfin, [insérez un élément HTML](#) pour le jeton.

**Une balise de début dont le nom de balise est l'un des suivants : "dd", "dt"**

Exécutez ces étapes :

1. Définissez le [drapeau frameset-ok](#) sur "pas ok".
2. Initialiser *le nœud* pour qu'il soit le [nœud actuel](#) (le nœud le plus bas de la pile).
3. *Boucle* : si *le nœud* est un [dd](#)élément, exécutez ces sous-étapes :
  1. [Générez des balises de fin implicites](#) , sauf pour [dd](#)les éléments.
  2. Si le [nœud actuel](#) n'est pas un [dd](#)élément, il s'agit d'une [erreur d'analyse](#) .
  3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [dd](#) élément ait été extrait de la pile.
  4. Passez à l'étape indiquée *ci* -dessous.
4. Si *node* est un [dt](#)élément, exécutez ces sous-étapes :
  1. [Générez des balises de fin implicites](#) , sauf pour [dt](#)les éléments.
  2. Si le [nœud actuel](#) n'est pas un [dt](#)élément, il s'agit d'une [erreur d'analyse](#) .
  3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [dt](#) élément ait été extrait de la pile.
  4. Passez à l'étape indiquée *ci* -dessous.
5. Si *le nœud* est dans la catégorie [spéciale](#) , mais n'est pas un élément [address](#), [div](#), ou [p](#), passez à l'étape intitulée *done* ci-dessous.
6. Sinon, définissez *node* sur l'entrée précédente dans la [pile d'éléments ouverts](#) et revenez à l'étape étiquetée *loop* .
7. *Terminé* : si la [pile d'éléments ouverts contient un pélément dans la portée du bouton](#) , [fermez un pélément](#) .
8. Enfin, [insérez un élément HTML](#) pour le jeton.

**Une balise de début dont le nom de balise est "texte clair"**

Si la [pile d'éléments ouverts](#) a un [p](#)élément dans la portée du bouton , alors [fermez un p élément](#) .

[Insérez un élément HTML](#) pour le jeton.

Passez le tokenizer à l' [état PLAINTEXT](#) .

*Une fois qu'une balise de début avec le nom de balise "plaintext" a été vue, ce sera le dernier jeton jamais vu autre que les jetons de caractère (et le jeton de fin de fichier), car il n'y a aucun moyen de sortir de l'état [PLAINTEXT](#) .*

#### Une balise de début dont le nom de balise est "bouton"

1. Si la [pile d'éléments ouverts](#) a un [button](#)élément dans scope , exécutez ces sous-étapes :
  1. [Erreur d'analyse](#) .
  2. [Générez des balises de fin implicites](#) .
  3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [button](#) élément ait été extrait de la pile.
2. [Reconstruire les éléments de formatage actifs](#) , le cas échéant.
3. [Insérez un élément HTML](#) pour le jeton.
4. Définissez le [drapeau frameset-ok](#) sur "pas ok".

#### Une balise de fin dont le nom de balise est l'un des suivants : "address", "article", "aside", "blockquote", "button", "center", "details", "dialog", "div", "dl", "fieldset", "figcaption", "figure", "footer", "header", "hgroup", "listing", "main", "menu", "nav", "ol", "pre ", "section", "résumé", "ul"

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée](#) qui soit un [élément HTML](#) avec le même nom de balise que celui du jeton, alors c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, exécutez ces étapes :

1. [Générez des balises de fin implicites](#) .
2. Si le [nœud actuel](#) n'est pas un [élément HTML](#) avec le même nom de balise que celui du jeton, il s'agit d'une [erreur d'analyse](#) .
3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [élément HTML](#) avec le même nom de balise que le jeton ait été extrait de la pile.

#### Une balise de fin dont le nom de balise est "form"

S'il n'y a pas [template](#)d'élément sur la [pile d'éléments ouverts](#) , exécutez ces sous-étapes :



1. Soit *node* l'élément sur lequel le [form pointeur d'élément](#) est défini, ou null s'il n'est pas défini sur un élément.
2. Définissez le [form pointeur d'élément](#) sur null.
3. Si *node* est null ou si la [pile d'éléments ouverts](#) n'a pas [de node dans la portée](#) , alors c'est une [erreur d'analyse](#) ; retourner et ignorer le jeton.
4. [Générez des balises de fin implicites](#) .
5. Si le [nœud actuel](#) n'est pas *node* , il s'agit d'une [erreur d'analyse](#) .
6. Supprimer le *nœud* de la [pile des éléments ouverts](#) .

S'il y a un [template](#) élément sur la [pile d'éléments ouverts](#) , exécutez plutôt ces sous-étapes :

1. Si la [pile d'éléments ouverts](#) n'a pas [d' form élément dans la portée](#) , alors c'est une [erreur d'analyse](#) ; retourner et ignorer le jeton.
2. [Générez des balises de fin implicites](#) .
3. Si le [nœud actuel](#) n'est pas un [form](#) élément, il s'agit d'une [erreur d'analyse](#) .
4. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [form](#) élément ait été extrait de la pile.

#### Une balise de fin dont le nom de balise est "p"

Si la [pile d'éléments ouverts](#) n'a pas [d' p élément dans la portée du bouton](#) , alors c'est une [erreur d'analyse](#) ; [insérez un élément HTML](#) pour un jeton de balise de début "p" sans attributs.

[Fermer un p élément](#) .

#### Une balise de fin dont le nom de balise est "li"

Si la [pile d'éléments ouverts](#) n'a pas [d' li élément dans la portée de l'élément de liste](#) , alors c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, exécutez ces étapes :

1. [Générez des balises de fin implicites](#) , sauf pour [li](#) les éléments.
2. Si le [nœud actuel](#) n'est pas un [li](#) élément, il s'agit d'une [erreur d'analyse](#) .
3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [li](#) élément ait été extrait de la pile.

#### Une balise de fin dont le nom de balise est l'un des suivants : "dd", "dt"

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée](#) qui soit un [élément HTML](#) avec le même nom de balise que celui du jeton, alors c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, exécutez ces étapes :

1. [Générez des balises de fin implicites](#) , sauf pour [les éléments HTML](#) avec le même nom de balise que le jeton.
2. Si le [nœud actuel](#) n'est pas un [élément HTML](#) avec le même nom de balise que celui du jeton, il s'agit d'une [erreur d'analyse](#) .
3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [élément HTML](#) avec le même nom de balise que le jeton ait été extrait de la pile.

**Une balise de fin dont le nom de balise est l'un des suivants : "h1", "h2", "h3", "h4", "h5", "h6"**

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée](#) qui est un [élément HTML](#) et dont le nom de balise est l'un des "h1", "h2", "h3", "h4", "h5" ou "h6", alors c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, exécutez ces étapes :

1. [Générez des balises de fin implicites](#) .
2. Si le [nœud actuel](#) n'est pas un [élément HTML](#) avec le même nom de balise que celui du jeton, il s'agit d'une [erreur d'analyse](#) .
3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [élément HTML](#) dont le nom de balise soit "h1", "h2", "h3", "h4", "h5" ou "h6" ait été extrait de la pile.

**Une balise de fin dont le nom de balise est "sarcasme"**

Respirez profondément, puis agissez comme décrit dans l'entrée "toute autre balise de fin" ci-dessous.

**Une balise de début dont le nom de balise est "a"**

Si la [liste des éléments de mise en forme actifs](#) contient un [a](#)élément entre la fin de la liste et le dernier [marqueur](#) de la liste (ou le début de la liste s'il n'y a pas [de marqueur](#) dans la liste), alors c'est une [erreur d'analyse](#) ; exécutez l' [algorithme de l'agence d'adoption](#) pour le jeton, puis supprimez cet élément de la [liste des éléments de mise en forme actifs](#) et de la [pile d'éléments ouverts](#) si l' [algorithme de l'agence d'adoption](#) ne l'a pas déjà supprimé (ce n'est peut-être pas le cas si l'élément n'est pas [dans la portée du tableau](#) ).

Dans le flux non conforme `<a href="a">a<table><a href="b">b</table>x`, le premier [a](#)élément serait fermé à la vue du second, et le caractère "x" serait à l'intérieur d'un lien vers "b", et non vers "a". Ceci malgré le fait que l' [a](#) élément externe n'est pas dans la portée du tableau (ce qui signifie qu'une `</a>` balise

de fin normale au début du tableau ne fermerait pas l' aélément externe). Le résultat est que les deux aéléments sont indirectement imbriqués l'un dans l'autre - un balisage non conforme entraînera souvent des DOM non conformes lors de l'analyse.

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton. [Poussez sur la liste des éléments de formatage actifs](#) cet élément.

**Une balise de début dont le nom de balise est l'un des suivants : "b", "big", "code", "em", "font", "i", "s", "small", "strike", "strong", "tt", "u"**

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton. [Poussez sur la liste des éléments de formatage actifs](#) cet élément.

**Une balise de début dont le nom de balise est "nobr"**

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

Si la [pile d'éléments ouverts a un nobrélément dans la portée](#) , alors c'est une [erreur d'analyse](#) ; exécutez l' [algorithme de l'agence d'adoption](#) pour le jeton, puis reconstruisez à nouveau [les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton. [Poussez sur la liste des éléments de formatage actifs](#) cet élément.

**Une balise de fin dont le nom de balise est l'un des suivants : "a", "b", "big", "code", "em", "font", "i", "nobr", "s", "small", "grève", "fort", "tt", "u"**

Exécutez l' [algorithme de l'agence d'adoption](#) pour le jeton.

**Une balise de début dont le nom de balise est l'un des suivants : "applet", "marquee", "object"**

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton.

Insérer un [marqueur](#) à la fin de la [liste des éléments de formatage actifs](#) .

Définissez le [drapeau frameset-ok](#) sur "pas ok".

**Un jeton de balise de fin dont le nom de balise est l'un des suivants : "applet", "marquee", "object"**

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée](#) qui soit un [élément HTML](#) avec le même nom de balise que celui du jeton, alors c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, exécutez ces étapes :

1. [Générez des balises de fin implicites](#) .
2. Si le [nœud actuel](#) n'est pas un [élément HTML](#) avec le même nom de balise que celui du jeton, il s'agit d'une [erreur d'analyse](#) .
3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [élément HTML](#) avec le même nom de balise que le jeton ait été extrait de la pile.
4. [Efface la liste des éléments de formatage actifs jusqu'au dernier marqueur](#) .

#### Une balise de début dont le nom de balise est "table"

Si le [Document](#) n'est pas défini sur [le mode Quirks](#) et que la [pile d'éléments ouverts](#) contient un [p](#)élément dans la portée du bouton , [fermez un p élément](#) .

[Insérez un élément HTML](#) pour le jeton.

Définissez le [drapeau frameset-ok](#) sur "pas ok".

Basculez le [mode d'insertion](#) sur " [dans le tableau](#) " .

#### Une balise de fin dont le nom de balise est "br"

[Erreur d'analyse](#) . Supprimez les attributs du jeton et agissez comme décrit dans l'entrée suivante ; c'est-à-dire agir comme s'il s'agissait d'un jeton de balise de début "br" sans attributs, plutôt que le jeton de balise de fin qu'il est en réalité.

#### Une balise de début dont le nom de balise est l'un des suivants : "area", "br", "embed", "img", "keygen", "wbr"

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaître l'indicateur de fermeture automatique du jeton](#) , s'il est défini.

Définissez le [drapeau frameset-ok](#) sur "pas ok".

#### Une balise de début dont le nom de balise est "input"

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaître l'indicateur de fermeture automatique du jeton](#) , s'il est défini.

Si le jeton n'a pas d'attribut avec le nom "type", ou s'il en a, mais que la valeur de cet attribut n'est pas une correspondance [ASCII insensible à la casse](#) pour la chaîne "hidden", alors : définissez l' [indicateur frameset-ok](#) sur "pas ok".

**Une balise de début dont le nom de balise est l'un des suivants : "param", "source", "track"**

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaitre l' indicateur de fermeture automatique du jeton](#) , s'il est défini.

**Une balise de début dont le nom de balise est "hr"**

Si la [pile d'éléments ouverts a un p élément dans la portée du bouton](#) , alors [fermez un p élément](#) .

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaitre l' indicateur de fermeture automatique du jeton](#) , s'il est défini.

Définissez le [drapeau frameset-ok](#) sur "pas ok".

**Une balise de début dont le nom de balise est "image"**

[Erreur d'analyse](#) . Remplacez le nom de balise du jeton par "img" et retraitez-le. (Ne demandez pas.)

**Une balise de début dont le nom de balise est "textarea"**

Exécutez ces étapes :

1. [Insérez un élément HTML](#) pour le jeton.
2. Si le [jeton suivant](#) est un jeton de caractère U+000A LINE FEED (LF), alors ignorez ce jeton et passez au suivant. (Les retours à la ligne au début des [textarea](#) éléments sont ignorés pour des raisons de commodité de création.)
3. Passez le tokenizer à l' [état RCDATA](#) .
4. Soit le [mode d'insertion d'origine](#) le [mode d'insertion](#) courant .
5. Définissez le [drapeau frameset-ok](#) sur "pas ok".
6. Basculez le [mode d'insertion](#) sur « [texte](#) ».

**Une balise de début dont le nom de balise est "xmp"**

Si la [pile d'éléments ouverts a un p élément dans la portée du bouton](#) , alors [fermez un p élément](#) .

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

Définissez le [drapeau frameset-ok](#) sur "pas ok".

Suivez l' [algorithme générique d'analyse d'éléments de texte brut](#) .

**Une balise de début dont le nom de balise est "iframe"**

Définissez le [drapeau frameset-ok](#) sur "pas ok".

Suivez l' [algorithme générique d'analyse d'éléments de texte brut](#) .

**Une balise de début dont le nom de balise est "noembed"**

**Une balise de début dont le nom de balise est "noscript", si l' [indicateur de script](#) est activé**

Suivez l' [algorithme générique d'analyse d'éléments de texte brut](#) .

**Une balise de début dont le nom de balise est "select"**

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton.

Définissez le [drapeau frameset-ok](#) sur "pas ok".

Si le [mode d'insertion](#) est l'un des " [dans le tableau](#) ", " [dans la légende](#) ", " [dans le corps du tableau](#) ", " [dans la ligne](#) " ou " [dans la cellule](#) ", basculez le [mode d'insertion](#) sur " [dans sélectionner dans le tableau](#) ". Sinon, basculez le [mode d'insertion](#) sur " [in select](#) ".

**Une balise de début dont le nom de balise est l'un des suivants : "optgroup", "option"**

Si le [nœud actuel](#) est un [option](#) élément, retirez le [nœud actuel](#) de la [pile des éléments ouverts](#) .

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton.

**Une balise de début dont le nom de balise est l'un des suivants : "rb", "rtc"**

Si la [pile d'éléments ouverts a un \[ruby\]\(#\)élément dans la portée](#) , alors [générez des balises de fin implicites](#) . Si le [nœud actuel](#) n'est plus un [ruby](#)élément, il s'agit d'une [erreur d'analyse](#) .

[Insérez un élément HTML](#) pour le jeton.

**Une balise de début dont le nom de balise est l'un des suivants : "rp", "rt"**

Si la [pile d'éléments ouverts a un \[ruby\]\(#\)élément dans scope](#) , alors [générez des balises de fin implicites](#) , sauf pour [rtc](#) les éléments. Si le [nœud actuel](#) n'est pas maintenant un [rtc](#) élément ou un [ruby](#)élément, il s'agit d'une [erreur d'analyse](#) .

[Insérez un élément HTML](#) pour le jeton.

#### Une balise de début dont le nom de balise est "math"

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Ajustez les attributs MathML](#) pour le jeton. (Cela corrige la casse des attributs MathML qui ne sont pas tous en minuscules.)

[Ajustez les attributs étrangers](#) pour le jeton. (Cela corrige l'utilisation des attributs d'espace de noms, en particulier XLink.)

[Insérez un élément étranger](#) pour le jeton, dans l' [espace de noms MathML](#) .

Si le jeton a son [indicateur de fermeture automatique](#) défini, retirez le [nœud actuel](#) de la [pile d'éléments ouverts](#) et reconnaissez l' [indicateur de fermeture automatique du jeton](#) .

#### Une balise de début dont le nom de balise est "svg"

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Ajustez les attributs SVG](#) pour le jeton. (Cela corrige la casse des attributs SVG qui ne sont pas tous en minuscules.)

[Ajustez les attributs étrangers](#) pour le jeton. (Cela corrige l'utilisation des attributs d'espace de noms, en particulier XLink dans SVG.)

[Insérez un élément étranger](#) pour le jeton, dans l' [espace de noms SVG](#) .

Si le jeton a son [indicateur de fermeture automatique](#) défini, retirez le [nœud actuel](#) de la [pile d'éléments ouverts](#) et reconnaissez l' [indicateur de fermeture automatique du jeton](#) .

#### Une balise de début dont le nom de balise est l'un des suivants : "caption", "col", "colgroup", "frame", "head", "tbody", "td", "tfoot", "th", "thead", "tr"

[Erreur d'analyse](#) . Ignorez le jeton.

#### Toute autre balise de début

[Reconstruire les éléments de formatage actifs](#) , le cas échéant.

[Insérez un élément HTML](#) pour le jeton.

*Cet élément sera un élément [ordinaire](#) .*

#### Toute autre balise de fin

Exécutez ces étapes :

1. Initialiser *le nœud* pour qu'il soit le [nœud actuel](#) (le nœud le plus bas de la pile).

2. *Boucle* : si le *nœud* est un [élément HTML](#) avec le même nom de balise que le jeton, alors :
  1. [Générez des balises de fin implicites](#) , sauf pour [les éléments HTML](#) avec le même nom de balise que le jeton.
  2. Si le *nœud* n'est pas le [nœud actuel](#) , il s'agit d'une [erreur d'analyse](#) .
  3. Faites éclater tous les nœuds du [nœud actuel](#) jusqu'au *nœud* , y compris le *nœud* , puis arrêtez ces étapes.
3. Sinon, si *node* est dans la catégorie [spéciale](#) , alors c'est une [erreur d'analyse](#) ; ignorez le jeton et retournez.
4. Définit *node* sur l'entrée précédente dans la [pile d'éléments ouverts](#) .
5. Revenez à l'étape intitulée *loop* .

Lorsque les étapes ci-dessus indiquent que l'agent utilisateur doit **fermer un pélément** , cela signifie que l'agent utilisateur doit exécuter les étapes suivantes :

1. [Générez des balises de fin implicites](#) , sauf pour [p](#) les éléments.
2. Si le [nœud actuel](#) n'est pas un [pélément](#), il s'agit d'une [erreur d'analyse](#) .
3. Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [pélément](#) ait été extrait de la pile.

L' **algorithme de l'agence d'adoption** , qui prend comme seul argument un jeton *pour* lequel l'algorithme est exécuté, se compose des étapes suivantes :

1. Soit *subject* le nom de la balise du *jeton* .
2. Si le [nœud actuel](#) est un [élément HTML](#) dont le nom de balise est *subject* , et que le [nœud actuel](#) n'est pas dans la [liste des éléments de mise en forme actifs](#) , retirez le [nœud actuel](#) de la [pile d'éléments ouverts](#) et revenez.
3. Laissez *le compteur de boucle externe* être 0.
4. Alors que c'est vrai :
  1. Si *le compteur de boucle externe* est supérieur ou égal à 8, alors retour.
  2. Incrémente *le compteur de boucle externe* de 1.
  3. Soit *élément de formatage* le dernier élément de la [liste des éléments de formatage actifs](#) qui :
    - est entre la fin de la liste et le dernier [marqueur](#) de la liste, le cas échéant, ou le début de la liste sinon, et



- a le nom de balise *subject* .

S'il n'y a pas un tel élément, retournez et agissez à la place comme décrit dans l'entrée "toute autre balise de fin" ci-dessus.

4. Si *l'élément de formatage* n'est pas dans la [pile des éléments ouverts](#) , alors c'est une [erreur d'analyse](#) ; supprimer l'élément de la liste et revenir.
5. Si *l'élément de formatage* est dans la [pile d'éléments ouverts](#) , mais que l'élément n'est pas [dans la portée](#) , alors c'est une [erreur d'analyse](#) ; retour.
6. Si *l'élément de formatage* n'est pas le [nœud actuel](#) , il s'agit d'une [erreur d'analyse](#) . (Mais ne reviens pas.)
7. Soit *le bloc le plus éloigné* le nœud le plus haut dans la [pile d'éléments ouverts](#) qui est plus bas dans la pile que *l'élément de formatage* et qui est un élément de la catégorie [spéciale](#) . Il n'y en a peut-être pas.
8. S'il n'y a pas *de bloc le plus éloigné* , alors l'UA doit d'abord faire apparaître tous les nœuds du bas de la [pile d'éléments ouverts](#) , du [nœud actuel](#) jusqu'à et y compris *l'élément de formatage* , puis supprimer *l'élément de formatage* de la [liste des éléments de formatage actifs](#) , et enfin revenir.
9. Soit *ancêtre commun* l'élément immédiatement au-dessus de *l'élément de formatage* dans la [pile d'éléments ouverts](#) .
10. Laissez un signet noter la position de *l'élément de formatage* dans la [liste des éléments de formatage actifs](#) par rapport aux éléments de chaque côté de celui-ci dans la liste.
11. Laissez *le nœud* et *le dernier nœud* être *le bloc le plus éloigné* .
12. Laissez *le compteur de boucle interne* être égal à 0.
13. Alors que c'est vrai :
  - Incrémente *le compteur de boucle interne* de 1.
  - Soit *node* l'élément immédiatement au-dessus de *node* dans la [pile d'éléments ouverts](#) , ou si *node* n'est plus dans la [pile d'éléments ouverts](#) (par exemple parce qu'il a été supprimé par cet algorithme), l'élément qui était immédiatement au-dessus de *node* dans la [pile d'éléments ouverts éléments](#) avant que *le nœud* ne soit supprimé.
  - Si *node* est un *élément de formatage* , alors [break](#) .

- Si le *compteur de boucle interne* est supérieur à 3 et que le *nœud* est dans la [liste des éléments de formatage actifs](#) , alors supprimez le *nœud* de la [liste des éléments de formatage actifs](#) .
  - Si *node* n'est pas dans la [liste des éléments de formatage actifs](#) , alors supprimez *node* de la [pile d'éléments ouverts](#) et [continuez](#) .
  - [Créez un élément pour le jeton](#) pour lequel le *nœud* d'élément a été créé, dans l' [espace de noms HTML](#) , avec un *ancêtre commun* comme parent prévu ; remplacez l'entrée pour *node* dans la [liste des éléments de formatage actifs](#) par une entrée pour le nouvel élément, remplacez l'entrée pour *node* dans la [pile d'éléments ouverts](#) par une entrée pour le nouvel élément, et laissez *node* être le nouvel élément.
  - Si le *dernier nœud* est le *bloc le plus éloigné* , déplacez le signet susmentionné pour qu'il soit immédiatement après le nouveau *nœud* dans la [liste des éléments de formatage actifs](#) .
  - [Ajouter](#) le *dernier nœud* au *nœud* .
  - Définissez le *dernier nœud* sur *node* .
14. Insérez le *dernier nœud* qui s'est retrouvé à l'étape précédente à l' [endroit approprié pour insérer un nœud](#) , mais en utilisant l'*ancêtre commun* comme *cible de remplacement* .
15. [Créez un élément pour le jeton](#) pour lequel l'*élément de mise en forme* a été créé, dans l' [espace de noms HTML](#) , avec le *bloc le plus éloigné* comme parent prévu.
16. Prenez tous les *nœuds enfants* du *bloc le plus éloigné* et ajoutez-les à l'élément créé à la dernière étape.
17. Ajoutez ce nouvel élément au *bloc le plus éloigné* .
18. Supprimez l'*élément de formatage* de la [liste des éléments de formatage actifs](#) et insérez le nouvel élément dans la [liste des éléments de formatage actifs](#) à la position du signet susmentionné.
19. Supprimez l'*élément de mise en forme* de la [pile d'éléments ouverts](#) et insérez le nouvel élément dans la [pile d'éléments ouverts](#) immédiatement en dessous de la position du *bloc le plus éloigné* de cette pile.

*Le nom de cet algorithme, "l'algorithme de l'agence d'adoption", vient de la façon dont il amène les éléments à changer de parents, et contraste avec [d'autres algorithmes possibles](#) pour traiter le contenu mal imbriqué.*

#### 13.2.6.4.8 Le mode d'insertion " texte "

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " text "](#) , l'agent utilisateur doit gérer le jeton comme suit :

##### Un jeton de personnage

[Insérez le caractère du jeton](#) .

*Il ne peut jamais s'agir d'un caractère U+0000 NULL ; le tokenizer les convertit en caractères U+FFFD REPLACEMENT CHARACTER.*

##### Un jeton de fin de fichier

[Erreur d'analyse](#) .

Si le [nœud actuel](#) est un [script](#) élément, définissez son [déjà commencé](#) sur vrai.

Détachez le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

Basculez le [mode d'insertion](#) sur le [mode d'insertion d'origine](#) et retraitez le jeton.

##### Une balise de fin dont le nom de balise est "script"

Si l' [analyseur HTML spéculatif actif](#) est nul et que la [pile de contexte d'exécution JavaScript](#) est vide, [effectuez un point de contrôle de microtâche](#) .

Soit *script* le [nœud courant](#) (qui sera un [script](#) élément).

Détachez le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

Basculez le [mode d'insertion](#) sur le [mode d'insertion d'origine](#) .

Laissez l' *ancien point d'insertion* avoir la même valeur que le [point d'insertion](#) actuel . Laissez le [point d'insertion](#) se trouver juste avant le [caractère d'entrée suivant](#) .

Incrémentez de un le [niveau d'imbrication des scripts de l'analyseur](#).

Si l' [analyseur HTML spéculatif actif](#) est nul, [préparez l'élément de script](#) *script* . Cela peut entraîner l'exécution de certains scripts, ce qui peut entraîner l' [insertion de nouveaux caractères dans le tokenizer](#) , et peut amener le tokenizer à générer plus de jetons, ce qui entraîne une [invocation réentrante de l'analyseur](#) .

Décrémentez le [niveau d'imbrication des scripts](#) de l'analyseur d'une unité. Si le [niveau d'imbrication du script](#) de l'analyseur est zéro, définissez l' [indicateur de pause de l'analyseur](#) sur faux.

Laissez au [point d'insertion](#) la valeur de l' *ancien point d'insertion* . (En d'autres termes, restaurez le [point d'insertion](#) à sa valeur précédente. Cette valeur peut être la valeur "indéfinie".)

À ce stade, si le [script de blocage d'analyse en attente](#) n'est pas nul, alors :

**Si le [niveau d'imbrication des scripts](#) n'est pas nul :**

Définissez l' [indicateur de pause de l'analyseur](#) sur true et annulez le traitement de toutes les invocations imbriquées du tokenizer, ce qui rend le contrôle à l'appelant. (La tokenisation reprendra lorsque l'appelant reviendra à l'étape de construction de l'arborescence "externe".)

*L'étape de construction de l'arborescence de cet analyseur particulier est [appelée de manière réentrante](#) , disons à partir d'un appel à `document.write()`.*

**Sinon:**

Tant que le [script de blocage d'analyse en attente](#) n'est pas nul :

1. Soit *le script* le [script de blocage d'analyse en attente](#) .
2. Définissez le [script de blocage d'analyse en attente](#) sur null.
3. [Démarrez l'analyseur HTML spéculatif](#) pour cette instance de l'analyseur HTML.
4. Bloquez le [tokenizer](#) pour cette instance de l' [analyseur HTML](#) , de sorte que la [boucle d'événements](#) n'exécute pas [les tâches](#) qui appellent le [tokenizer](#) .
5. Si l'analyseur `Document` [a une feuille de style qui bloque les scripts](#) ou si *le script* est [prêt à être exécuté par l'analyseur](#) est faux : [faites tourner la boucle d'événements](#) jusqu'à ce que l'analyseur `Document` [n'ait plus de feuille de style qui bloque les scripts](#) et que *le script* soit [prêt à être analysé. exécuté](#) devient vrai.
6. Si cet [analyseur a été interrompu](#) entre-temps, retournez.

*Cela peut se produire si, par exemple, pendant l' [exécution de l'algorithme de la boucle d'événements](#)`Document` , le est [détruit](#) ou la `document.open()` [méthode est invoquée](#) sur le `Document`.*

7. [Arrêtez l'analyseur HTML spéculatif](#) pour cette instance de l'analyseur HTML.
8. Débloquez le [tokenizer](#) pour cette instance de l' [analyseur HTML](#) , de sorte que [les tâches](#) qui appellent le [tokenizer](#) puissent à nouveau être exécutées.
9. Laissez le [point d'insertion](#) se trouver juste avant le [caractère d'entrée suivant](#) .

10. Incrémente le [niveau d'imbrication des scripts](#) de l'analyseur de un (il doit être égal à zéro avant cette étape, donc cela le définit sur un).
11. [Exécutez l'élément de script](#) le script .
12. Décrémente le [niveau d'imbrication des scripts](#) de l'analyseur d'une unité. Si le [niveau d'imbrication du script](#) de l'analyseur est égal à zéro (ce qui devrait toujours être le cas à ce stade), définissez l' [indicateur de pause de l'analyseur](#) sur false.
13. Laissez à nouveau le [point d'insertion](#) indéfini.

#### Toute autre balise de fin

Détachez le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

Basculez le [mode d'insertion](#) sur le [mode d'insertion d'origine](#) .

#### 13.2.6.4.9 Le mode d'insertion " dans tableau "

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " dans la table "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère, si le [nœud actuel](#) est [table](#), [tbody](#), [template](#), [tfoot](#), [thead](#), ou [tr](#) élément**

Laissez les *jetons de caractère de table en attente* être une liste vide de jetons.

Soit le [mode d'insertion d'origine](#) le [mode d'insertion](#) courant .

Basculez le [mode d'insertion](#) sur " [dans le texte du tableau](#) " et retraitez le jeton.

#### Un jeton de commentaire

[Insérez un commentaire](#) .

#### Un jeton DOCTYPE

[Erreur d'analyse](#) . Ignorez le jeton.

#### Une balise de début dont le nom de balise est "légende"

[Efface la pile vers un contexte de table](#) . (Voir ci-dessous.)

Insérer un [marqueur](#) à la fin de la [liste des éléments de formatage actifs](#) .

[Insérez un élément HTML](#) pour le token, puis basculez le [mode d'insertion](#) sur " [in caption](#) ".

**Une balise de début dont le nom de balise est "colgroup"**

[Efface la pile vers un contexte de table](#) . (Voir ci-dessous.)

[Insérez un élément HTML](#) pour le jeton, puis basculez le [mode d'insertion](#) sur "[dans le groupe de colonnes](#)".

**Une balise de début dont le nom de balise est "col"**

[Efface la pile vers un contexte de table](#) . (Voir ci-dessous.)

[Insérez un élément HTML](#) pour un jeton de balise de début "colgroup" sans attribut, puis basculez le [mode d'insertion](#) sur "[dans le groupe de colonnes](#)".

Retraitez le jeton actuel.

**Une balise de début dont le nom de balise est l'un des suivants : "tbody", "tfoot", "thead"**

[Efface la pile vers un contexte de table](#) . (Voir ci-dessous.)

[Insérez un élément HTML](#) pour le jeton, puis basculez le [mode d'insertion](#) sur "[dans le corps du tableau](#)".

**Une balise de début dont le nom de balise est l'un des suivants : "td", "th", "tr"**

[Efface la pile vers un contexte de table](#) . (Voir ci-dessous.)

[Insérez un élément HTML](#) pour un jeton de balise de début "tbody" sans attribut, puis basculez le [mode d'insertion](#) sur "[dans le corps du tableau](#)".

Retraitez le jeton actuel.

**Une balise de début dont le nom de balise est "table"**

[Erreur d'analyse](#) .

Si la [pile d'éléments ouverts](#) n'a pas d'[table](#)élément dans la portée de la [table](#) , ignorez le jeton.

Sinon:

Extraire des éléments de cette pile jusqu'à ce qu'un [table](#)élément ait été extrait de la pile.

[Réinitialisez le mode d'insertion de manière appropriée](#) .

Retraitez le jeton.

**Une balise de fin dont le nom de balise est "table"**

Si la [pile d'éléments ouverts](#) n'a pas d'[table](#)élément dans la portée de la [table](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon:

Extraire des éléments de cette pile jusqu'à ce qu'un tableélément ait été extrait de la pile.

Réinitialisez le mode d'insertion de manière appropriée .

**Une balise de fin dont le nom de balise est l'un des suivants : "body", "caption", "col", "colgroup", "html", "tbody", "td", "tfoot", "th", "thead", "tr"**

Erreur d'analyse . Ignorez le jeton.

**Une balise de début dont le nom de balise est l'un des suivants : "style", "script", "template"**

**Une balise de fin dont le nom de balise est "template"**

Traiter le jeton en suivant les règles du mode d'insertion " in head " .

**Une balise de début dont le nom de balise est "input"**

Si le jeton n'a pas d'attribut avec le nom "type", ou s'il en a, mais que la valeur de cet attribut n'est pas une correspondance ASCII insensible à la casse pour la chaîne " `hidden`", alors : agissez comme décrit dans l'entrée "autre chose" dessous.

Sinon:

Erreur d'analyse .

Insérez un élément HTML pour le jeton.

Retirez cet inputélément de la pile d'éléments ouverts .

Reconnaitre l'indicateur de fermeture automatique du jeton , s'il est défini.

**Une balise de début dont le nom de balise est "form"**

Erreur d'analyse .

S'il y a un templateélément sur la pile d'éléments ouverts , ou si le formpointeur d'élément n'est pas nul, ignorez le jeton.

Sinon:

Insérez un élément HTML pour le jeton et définissez le formpointeur d'élément pour qu'il pointe vers l'élément créé.

Retirez cet formélément de la pile d'éléments ouverts .

**Un jeton de fin de fichier**

Traitez le jeton en utilisant les règles du mode d'insertion « dans le corps » .

## Rien d'autre

[Erreur d'analyse](#) . Activez [le parentage adoptif](#) , traitez le jeton [en utilisant les règles du mode d'insertion « dans le corps »](#) , puis désactivez [le parentage nourricier](#) .

Lorsque les étapes ci-dessus nécessitent que l'UA **réinitialise la pile dans un contexte de table** , cela signifie que l'UA doit, tant que le [nœud actuel](#) n'est pas un élément [table](#), [template](#) ou [html](#), extraire des éléments de la [pile d'éléments ouverts](#) .

*Il s'agit de la même liste d'éléments que celle utilisée dans les étapes [contient un élément dans la portée de la table](#) .*

*Le [nœud courant](#) étant un [html](#) élément après ce processus est un [cas de fragment](#) .*

### 13.2.6.4.10 Le mode d'insertion " dans le texte du tableau "

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " dans le texte du tableau](#) " , l'agent utilisateur doit gérer le jeton comme suit :

#### Un jeton de caractère qui est U+0000 NULL

[Erreur d'analyse](#) . Ignorez le jeton.

#### Tout autre jeton de personnage

Ajoutez le jeton de personnage à la liste [des jetons de personnage de la table en attente](#) .

## Rien d'autre

Si l'un des jetons de la liste [des jetons de caractères de la table en attente](#) sont des jetons de caractères qui ne sont pas [des espaces blancs ASCII](#) , il s'agit d'une [erreur d'analyse](#) : retirez les jetons de caractères dans la liste [des jetons de caractères de la table en attente](#) en utilisant les règles données dans l'entrée "autre chose" en mode d'insertion " [dans tableau](#) " .

Sinon, [insérez les caractères](#) donnés par la liste [des jetons de caractères de la table en attente](#) .

Basculez le [mode d'insertion](#) sur le [mode d'insertion d'origine](#) et retirez le jeton.

### 13.2.6.4.11 Le mode d'insertion " en légende "

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion « in caption »](#) , l'agent utilisateur doit gérer le jeton comme suit :



### Une balise de fin dont le nom de balise est "légende"

Si la [pile d'éléments ouverts](#) n'a pas d' [caption](#)élément dans la portée de la [table](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton. ( [cas de fragment](#) )

Sinon:

[Générez des balises de fin implicites](#) .

Maintenant, si le [nœud actuel](#) n'est pas un [caption](#)élément, il s'agit d'une [erreur d'analyse](#) .

Extraire des éléments de cette pile jusqu'à ce qu'un [caption](#)élément ait été extrait de la pile.

[Efface la liste des éléments de formatage actifs jusqu'au dernier marqueur](#) .

Basculez le [mode d'insertion](#) sur " [dans le tableau](#) ".

### Une balise de début dont le nom de balise est l'un des suivants : "caption", "col", "colgroup", "tbody", "td", "tfoot", "th", "thead", "tr"

#### Une balise de fin dont le nom de balise est "table"

Si la [pile d'éléments ouverts](#) n'a pas d' [caption](#)élément dans la portée de la [table](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton. ( [cas de fragment](#) )

Sinon:

[Générez des balises de fin implicites](#) .

Maintenant, si le [nœud actuel](#) n'est pas un [caption](#)élément, il s'agit d'une [erreur d'analyse](#) .

Extraire des éléments de cette pile jusqu'à ce qu'un [caption](#)élément ait été extrait de la pile.

[Efface la liste des éléments de formatage actifs jusqu'au dernier marqueur](#) .

Basculez le [mode d'insertion](#) sur " [dans le tableau](#) ".

Retraitez le jeton.

### Une balise de fin dont le nom de balise est l'un des suivants : "body", "col", "colgroup", "html", "tbody", "td", "tfoot", "th", "thead", "tr"

[Erreur d'analyse](#) . Ignorez le jeton.

### Rien d'autre

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

#### **13.2.6.4.12 Le mode d'insertion " dans groupe de colonnes "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " dans le groupe de colonnes "](#), l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

[Insérez le caractère](#) .

**Un jeton de commentaire**

[Insérez un commentaire](#) .

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Une balise de début dont le nom de balise est "col"**

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaitre l'indicateur de fermeture automatique du jeton](#) , s'il est défini.

**Une balise de fin dont le nom de balise est "colgroup"**

Si le [nœud courant](#) n'est pas un [colgroup](#)élément, alors c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, pop le [nœud actuel](#) de la [pile d'éléments ouverts](#) . Basculez le [mode d'insertion](#) sur " [dans le tableau](#) " .

**Une balise de fin dont le nom de balise est "col"**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "template"**

**Une balise de fin dont le nom de balise est "template"**

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [in head](#) " .

**Un jeton de fin de fichier**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Rien d'autre**

Si le [nœud courant](#) n'est pas un [colgroup](#)élément, alors c'est une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, pop le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

Basculez le [mode d'insertion](#) sur " [dans le tableau](#) ".

Retraitez le jeton.

#### **13.2.6.4.13 Le mode d'insertion " dans le corps du tableau "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " dans le corps de la table](#) ", l'agent utilisateur doit gérer le jeton comme suit :

**Une balise de début dont le nom de balise est "tr"**

[Efface la pile vers un contexte de corps de table](#) . (Voir ci-dessous.)

[Insérez un élément HTML](#) pour le token, puis basculez le [mode d'insertion](#) sur " [in row](#) ".

**Une balise de début dont le nom de balise est l'un des suivants : "th", "td"**

[Erreur d'analyse](#) .

[Efface la pile vers un contexte de corps de table](#) . (Voir ci-dessous.)

[Insérez un élément HTML](#) pour un jeton de balise de début "tr" sans attribut, puis passez le [mode d'insertion](#) sur " [dans la ligne](#) ".

Retraitez le jeton actuel.

**Une balise de fin dont le nom de balise est l'un des suivants : "tbody", "tfoot", "thead"**

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée du tableau](#) qui soit un [élément HTML](#) avec le même nom de balise que le jeton, il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon:

[Efface la pile vers un contexte de corps de table](#) . (Voir ci-dessous.)

Pop le [nœud actuel](#) de la [pile d'éléments ouverts](#) . Basculez le [mode d'insertion](#) sur " [dans le tableau](#) ".

**Une balise de début dont le nom de balise est l'un des suivants : "caption", "col", "colgroup", "tbody", "tfoot", "thead"**

**Une balise de fin dont le nom de balise est "table"**

Si la [pile d'éléments ouverts](#) n'a pas [d' tbodyélément, thead ou tfoot dans l'étendue de la table](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon:

[Efface la pile vers un contexte de corps de table](#) . (Voir ci-dessous.)

Pop le [nœud actuel](#) de la [pile d'éléments ouverts](#) . Basculez le [mode d'insertion](#) sur " [dans le tableau](#) " .

Retraitez le jeton.

**Une balise de fin dont le nom de balise est l'un des suivants : "body", "caption", "col", "colgroup", "html", "td", "th", "tr"**

[Erreur d'analyse](#) . Ignorez le jeton.

**Rien d'autre**

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [dans la table](#) " .

Lorsque les étapes ci-dessus nécessitent que l'agent utilisateur **efface la pile dans un contexte de corps de table** , cela signifie que l'agent utilisateur doit, tant que le [nœud actuel](#) n'est pas un élément [tbody](#), [tfoot](#), [thead](#), [template](#) ou [html](#) élément, extraire les éléments de la [pile des éléments ouverts](#) .

*Le [nœud courant](#) étant un [html](#) élément après ce processus est un [cas de fragment](#) .*

#### **13.2.6.4.14 Le mode d'insertion " en ligne "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " en ligne "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Une balise de début dont le nom de balise est l'un des suivants : "th", "td"**

[Efface la pile vers un contexte de ligne de table](#) . (Voir ci-dessous.)

[Insérez un élément HTML](#) pour le jeton, puis basculez le [mode d'insertion](#) sur " [dans la cellule](#) " .

Insérer un [marqueur](#) à la fin de la [liste des éléments de formatage actifs](#) .

**Une balise de fin dont le nom de balise est "tr"**

Si la [pile d'éléments ouverts](#) n'a pas d' [tr](#) élément dans la portée de la table , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon:

[Efface la pile vers un contexte de ligne de table](#) . (Voir ci-dessous.)

Pop le [nœud actuel](#) (qui sera un [tr](#) élément) de la [pile d'éléments ouverts](#) . Basculez le [mode d'insertion](#) sur " [dans le corps du tableau](#) " .

**Une balise de début dont le nom de balise est l'un des suivants : "caption", "col", "colgroup", "tbody", "tfoot", "thead", "tr"**

**Une balise de fin dont le nom de balise est "table"**

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée de la table](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon:

[Efface la pile vers un contexte de ligne de table](#) . (Voir ci-dessous.)

Pop le [nœud actuel](#) (qui sera un [tr](#)élément) de la [pile d'éléments ouverts](#) . Basculez le [mode d'insertion](#) sur " [dans le corps du tableau](#) " .

Retraitez le jeton.

**Une balise de fin dont le nom de balise est l'un des suivants : "tbody", "tfoot", "thead"**

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée du tableau](#) qui soit un [élément HTML](#) avec le même nom de balise que le jeton, il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton.

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée de la table](#) , ignorez le jeton.

Sinon:

[Efface la pile vers un contexte de ligne de table](#) . (Voir ci-dessous.)

Pop le [nœud actuel](#) (qui sera un [tr](#)élément) de la [pile d'éléments ouverts](#) . Basculez le [mode d'insertion](#) sur " [dans le corps du tableau](#) " .

Retraitez le jeton.

**Une balise de fin dont le nom de balise est l'un des suivants : "body", "caption", "col", "colgroup", "html", "td", "th"**

[Erreur d'analyse](#) . Ignorez le jeton.

**Rien d'autre**

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [dans la table](#) " .

Lorsque les étapes ci-dessus demandent à l'UA d' **effacer la pile dans un contexte de ligne de table** , cela signifie que l'UA doit, tant que le [nœud actuel](#) n'est pas un élément [tr](#), [template](#) ou [html](#), extraire des éléments de la [pile d'éléments ouverts](#) .

*Le [nœud courant](#) étant un [html](#) élément après ce processus est un [cas de fragment](#) .*

#### 13.2.6.4.15 Le mode d'insertion " dans la cellule "

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " dans la cellule "](#), l'agent utilisateur doit gérer le jeton comme suit :

##### Une balise de fin dont le nom de balise est l'un des suivants : "td", "th"

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée du tableau](#) qui soit un [élément HTML](#) avec le même nom de balise que celui du jeton, il s'agit alors d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon:

[Générez des balises de fin implicites](#) .

Désormais, si le [nœud actuel](#) n'est pas un [élément HTML](#) avec le même nom de balise que le jeton, il s'agit d'une [erreur d'analyse](#) .

Extraire les éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [élément HTML](#) avec le même nom de balise que le jeton ait été extrait de la pile.

[Efface la liste des éléments de formatage actifs jusqu'au dernier marqueur](#) .

Basculez le [mode d'insertion](#) sur " [en ligne](#) " .

##### Une balise de début dont le nom de balise est l'un des suivants : "caption", "col", "colgroup", "tbody", "td", "tfoot", "th", "thead", "tr"

Si la [pile d'éléments ouverts](#) n'a pas [d'élément td ou th dans la portée de la table](#), alors c'est une [erreur d'analyse](#) ; ignorer le jeton. ( [cas de fragment](#) )

Sinon, [fermez la cellule](#) (voir ci-dessous) et retraitez le jeton.

##### Une balise de fin dont le nom de balise est l'un des suivants : "body", "caption", "col", "colgroup", "html"

[Erreur d'analyse](#) . Ignorez le jeton.

##### Une balise de fin dont le nom de balise est l'un des suivants : "table", "tbody", "tfoot", "thead", "tr"

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée du tableau](#) qui soit un [élément HTML](#) avec le même nom de balise que celui du jeton, il s'agit alors d'une [erreur d'analyse](#) ; ignorer le jeton.

Sinon, [fermez la cellule](#) (voir ci-dessous) et retraitez le jeton.

##### Rien d'autre

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

Lorsque les étapes ci-dessus indiquent de **fermer la cellule**, elles signifient exécuter l'algorithme suivant :

1. [Générez des balises de fin implicites](#) .
2. Si le [nœud actuel](#) n'est pas maintenant un [td](#)élément ou un [th](#) élément, il s'agit d'une [erreur d'analyse](#) .
3. Extraire les éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [td](#) élément ou un [th](#)élément ait été extrait de la pile.
4. [Efface la liste des éléments de formatage actifs jusqu'au dernier marqueur](#) .
5. Basculez le [mode d'insertion](#) sur " [en ligne](#) " .

*La [pile d'éléments ouverts](#) ne peut pas avoir à la fois un élément [td](#) et un [th](#)élément [dans la portée de la table](#) , ni l'un ni l'autre lorsque l' algorithme [de fermeture de la cellule](#) est invoqué.*

#### **13.2.6.4.16 Le mode d'insertion « in select »**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " in select "](#) , l'agent utilisateur doit gérer le jeton comme suit :

##### **Un jeton de caractère qui est U+0000 NULL**

[Erreur d'analyse](#) . Ignorez le jeton.

##### **Tout autre jeton de personnage**

[Insérez le caractère du jeton](#) .

##### **Un jeton de commentaire**

[Insérez un commentaire](#) .

##### **Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

##### **Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

##### **Une balise de début dont le nom de balise est "option"**

Si le [nœud actuel](#) est un [option](#)élément, sortez ce nœud de la [pile des éléments ouverts](#) .

[Insérez un élément HTML](#) pour le jeton.

##### **Une balise de début dont le nom de balise est "optgroup"**

Si le [nœud actuel](#) est un [option](#)élément, sortez ce nœud de la [pile des éléments ouverts](#) .

Si le [nœud actuel](#) est un [optgroup](#)élément, sortez ce nœud de la [pile des éléments ouverts](#) .

[Insérez un élément HTML](#) pour le jeton.

#### Une balise de fin dont le nom de balise est "optgroup"

Tout d'abord, si le [nœud actuel](#) est un [option](#)élément et que le nœud qui le précède immédiatement dans la [pile d'éléments ouverts](#) est un [optgroup](#) élément, alors extrayez le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

Si le [nœud actuel](#) est un [optgroup](#)élément, alors sortez ce nœud de la [pile des éléments ouverts](#) . Sinon, c'est une [erreur d'analyse](#) ; ignorer le jeton.

#### Une balise de fin dont le nom de balise est "option"

Si le [nœud actuel](#) est un [option](#)élément, alors sortez ce nœud de la [pile des éléments ouverts](#) . Sinon, c'est une [erreur d'analyse](#) ; ignorer le jeton.

#### Une balise de fin dont le nom de balise est "select"

Si la [pile d'éléments ouverts](#) n'a pas d' [select](#)élément dans la portée de [sélection](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton. ( [cas de fragment](#) )

Sinon:

Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [select](#)élément ait été extrait de la pile.

[Réinitialisez le mode d'insertion de manière appropriée](#) .

#### Une balise de début dont le nom de balise est "select"

[Erreur d'analyse](#) .

Si la [pile d'éléments ouverts](#) n'a pas d' [select](#)élément dans [select scope](#) , ignorez le jeton. ( [cas de fragment](#) )

Sinon:

Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [select](#)élément ait été extrait de la pile.

[Réinitialisez le mode d'insertion de manière appropriée](#) .

*Il est simplement traité comme une balise de fin.*

#### Une balise de début dont le nom de balise est l'un des suivants : "input", "keygen", "textarea"

[Erreur d'analyse](#) .



Si la [pile d'éléments ouverts](#) n'a pas d' [select](#) élément dans [select scope](#) , ignorez le jeton. ( [cas de fragment](#) )

Sinon:

Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [select](#) élément ait été extrait de la pile.

[Réinitialisez le mode d'insertion de manière appropriée](#) .

Retraitez le jeton.

**Une balise de début dont le nom de balise est l'un des suivants : "script", "template"**

**Une balise de fin dont le nom de balise est "template"**

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [in head](#) " .

**Un jeton de fin de fichier**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Rien d'autre**

[Erreur d'analyse](#) . Ignorez le jeton.

#### **13.2.6.4.17 Le mode d'insertion " in select in table "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion « in select in table »](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Une balise de début dont le nom de balise est l'un des suivants : "caption", "table", "tbody", "tfoot", "thead", "tr", "td", "th"**

[Erreur d'analyse](#) .

Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [select](#) élément ait été extrait de la pile.

[Réinitialisez le mode d'insertion de manière appropriée](#) .

Retraitez le jeton.

**Une balise de fin dont le nom de balise est l'un des suivants : "caption", "table", "tbody", "tfoot", "thead", "tr", "td", "th"**

[Erreur d'analyse](#) .

Si la [pile d'éléments ouverts](#) n'a pas [d'élément dans la portée du tableau](#) qui soit un [élément HTML](#) avec le même nom de balise que celui du jeton, ignorez le jeton.

Sinon:

Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [select](#) élément ait été extrait de la pile.

[Réinitialisez le mode d'insertion de manière appropriée](#) .

Retraitez le jeton.

#### **Rien d'autre**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [in select](#) » .

#### **13.2.6.4.18 Le mode d'insertion " dans modèle "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " dans le modèle "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de personnage**

**Un jeton de commentaire**

**Un jeton DOCTYPE**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Une balise de début dont le nom de balise est l'un des suivants : "base", "basefont", "bgsound", "link", "meta", "noframes", "script", "style", "template", "title"**

**Une balise de fin dont le nom de balise est "template"**

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [in head](#) " .

**Une balise de début dont le nom de balise est l'un des suivants : "caption", "colgroup", "tbody", "tfoot", "thead"**

Retirez le [mode d'insertion de modèle actuel](#) de la [pile des modes d'insertion de modèle](#) .

Poussez « [dans le tableau](#) » sur la [pile des modes d'insertion de modèle](#) afin qu'il s'agisse du nouveau [mode d'insertion de modèle actuel](#) .

Basculez le [mode d'insertion](#) sur " [dans table](#) ", et retraitez le jeton.

**Une balise de début dont le nom de balise est "col"**

Retirez le [mode d'insertion de modèle actuel](#) de la [pile des modes d'insertion de modèle](#) .

Poussez « [dans le groupe de colonnes](#) » sur la [pile des modes d'insertion de modèles](#) afin qu'il s'agisse du nouveau [mode d'insertion de modèle actuel](#) .

Basculez le [mode d'insertion](#) sur " [dans le groupe de colonnes](#) ", et retraits le jeton.

#### **Une balise de début dont le nom de balise est "tr"**

Retirez le [mode d'insertion de modèle actuel](#) de la [pile des modes d'insertion de modèle](#) .

Poussez « [dans le corps du tableau](#) » sur la [pile des modes d'insertion de modèle](#) afin qu'il soit le nouveau [mode d'insertion de modèle actuel](#) .

Basculez le [mode d'insertion](#) sur " [dans le corps du tableau](#) ", et retraits le jeton.

#### **Une balise de début dont le nom de balise est l'un des suivants : "td", "th"**

Retirez le [mode d'insertion de modèle actuel](#) de la [pile des modes d'insertion de modèle](#) .

Poussez « [dans la ligne](#) » sur la [pile des modes d'insertion de modèle](#) afin qu'il soit le nouveau [mode d'insertion de modèle actuel](#) .

Basculez le [mode d'insertion](#) sur " [in row](#) ", et retraits le jeton.

#### **Toute autre balise de début**

Retirez le [mode d'insertion de modèle actuel](#) de la [pile des modes d'insertion de modèle](#) .

Poussez " [dans le corps](#) " sur la [pile des modes d'insertion de modèle](#) afin qu'il soit le nouveau [mode d'insertion de modèle actuel](#) .

Basculez le [mode d'insertion](#) sur " [in body](#) ", et retraits le jeton.

#### **Toute autre balise de fin**

[Erreur d'analyse](#) . Ignorez le jeton.

#### **Un jeton de fin de fichier**

S'il n'y a pas [template](#) d'élément sur la [pile d'éléments ouverts](#) , alors [arrêtez l'analyse](#) . ( [cas de fragment](#) )

Sinon, il s'agit d'une [erreur d'analyse](#) .

Extraire des éléments de la [pile d'éléments ouverts](#) jusqu'à ce qu'un [template](#) élément ait été extrait de la pile.

[Efface la liste des éléments de formatage actifs jusqu'au dernier marqueur](#) .

Retirez le [mode d'insertion de modèle actuel](#) de la [pile des modes d'insertion de modèle](#) .

[Réinitialisez le mode d'insertion de manière appropriée](#) .

Retraitez le jeton.

#### **13.2.6.4.19 Le mode d'insertion " après le corps "**

Lorsque l'agent utilisateur doit appliquer les règles pour le [mode d'insertion " après le corps "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Un jeton de commentaire**

[Insérez un commentaire](#) comme dernier enfant du premier élément dans la [pile d'éléments ouverts](#) (l' [html](#) élément).

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Une balise de fin dont le nom de balise est "html"**

Si l'analyseur a été créé dans le cadre de l' [algorithme d'analyse de fragment HTML](#) , il s'agit d'une [erreur d'analyse](#) ; ignorer le jeton. ( [cas de fragment](#) )

Sinon, basculez le [mode d'insertion](#) sur " [après après corps](#) " .

**Un jeton de fin de fichier**

[Arrêtez l'analyse](#) .

**Rien d'autre**

[Erreur d'analyse](#) . Basculez le [mode d'insertion](#) sur " [in body](#) " et retraitez le jeton.

#### **13.2.6.4.20 Le mode d'insertion " in frameset "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " in frameset "](#), l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

[Insérez le caractère](#) .

**Un jeton de commentaire**

[Insérez un commentaire](#) .

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Une balise de début dont le nom de balise est "frameset"**

[Insérez un élément HTML](#) pour le jeton.

**Une balise de fin dont le nom de balise est "frameset"**

Si le [nœud actuel](#) est l'élément racine [html](#), alors c'est une [erreur d'analyse](#) ; ignorer le jeton. ( [cas de fragment](#) )

Sinon, pop le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

Si l'analyseur n'a pas été créé dans le cadre de l' [algorithme d'analyse de fragment HTML](#) ( [fragment case](#) ) et que le [nœud actuel](#) n'est plus un [frameset](#) élément, basculez le [mode d'insertion](#) sur " [après frameset](#) " .

**Une balise de début dont le nom de balise est "frame"**

[Insérez un élément HTML](#) pour le jeton. Retirez immédiatement le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

[Reconnaitre l' indicateur de fermeture automatique du jeton](#) , s'il est défini.

**Une balise de début dont le nom de balise est "noframes"**

Traiter le jeton [en suivant les règles du mode d'insertion " in head "](#) .

**Un jeton de fin de fichier**

Si le [nœud actuel](#) n'est pas l' [html](#) élément racine, il s'agit d'une [erreur d'analyse](#) .

*Le [nœud courant](#) ne peut être que l' [html](#) élément racine dans le [cas du fragment](#) .*

[Arrêtez l'analyse](#) .

Rien d'autre

[Erreur d'analyse](#) . Ignorez le jeton.

#### **13.2.6.4.21 Le mode d'insertion " après frameset "**

Lorsque l'agent utilisateur doit appliquer les règles pour le [mode d'insertion " après le jeu de cadres "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

[Insérez le caractère](#) .

**Un jeton de commentaire**

[Insérez un commentaire](#) .

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Une balise de fin dont le nom de balise est "html"**

Basculez le [mode d'insertion](#) sur " [après après frameset](#) " .

**Une balise de début dont le nom de balise est "noframes"**

Traiter le jeton [en suivant les règles du mode d'insertion](#) " [in head](#) " .

**Un jeton de fin de fichier**

[Arrêtez l'analyse](#) .

Rien d'autre

[Erreur d'analyse](#) . Ignorez le jeton.

#### **13.2.6.4.22 Le mode d'insertion " après après corps "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion « after after body »](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de commentaire**

[Insérez un commentaire](#) comme dernier enfant de l' [Document](#) objet.

**Un jeton DOCTYPE**

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Un jeton de fin de fichier**

[Arrêtez l'analyse](#) .

**Rien d'autre**

[Erreur d'analyse](#) . Basculez le [mode d'insertion](#) sur " [in body](#) " et retraitez le jeton.

#### **13.2.6.4.23 Le mode d'insertion " après après frameset "**

Lorsque l'agent utilisateur doit appliquer les règles du [mode d'insertion " après après le jeu de cadres "](#) , l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de commentaire**

[Insérez un commentaire](#) comme dernier enfant de l' [Document](#) objet.

**Un jeton DOCTYPE**

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

**Une balise de début dont le nom de balise est "html"**

Traitez le jeton [en utilisant les règles du mode d'insertion](#) « [dans le corps](#) » .

**Un jeton de fin de fichier**

[Arrêtez l'analyse](#) .

**Une balise de début dont le nom de balise est "noframes"**

Traiter le jeton [en suivant les règles du mode d'insertion " in head "](#) .

**Rien d'autre**

[Erreur d'analyse](#) . Ignorez le jeton.

#### **13.2.6.5 Les règles d'analyse des jetons dans le contenu étranger**

Lorsque l'agent utilisateur doit appliquer les règles d'analyse des jetons dans le contenu étranger, l'agent utilisateur doit gérer le jeton comme suit :

**Un jeton de caractère qui est U+0000 NULL**

[Erreur d'analyse](#) . [Insérez un caractère de REMPLACEMENT U+FFFD](#) .

**Un jeton de caractère parmi U+0009 CHARACTER TABULATION, U+000A LINE FEED (LF), U+000C FORM FEED (FF), U+000D CARRIAGE RETURN (CR) ou U+0020 SPACE**

[Insérez le caractère du jeton](#) .

**Tout autre jeton de personnage**

[Insérez le caractère du jeton](#) .

Définissez le [drapeau frameset-ok](#) sur "pas ok".

**Un jeton de commentaire**

[Insérez un commentaire](#) .

**Un jeton DOCTYPE**

[Erreur d'analyse](#) . Ignorez le jeton.

**Une balise de début dont le nom de balise est l'un des suivants :** "b", "big", "blockquote", "body", "br", "center", "code", "dd", "div", "dl", "dt", "em", "embed", "h1", "h2", "h3", "h4", "h5", "h6", "head", "hr", "i", "img", "li", "listing", "menu", "meta", "nobr", "ol", "p", "pre", "ruby", "s", "small", "span", "strong", "strike", "sub", "sup", "table", "tt", "u", "ul", "var"

**Une balise de début dont le nom de balise est "font", si le jeton a des attributs nommés "color", "face" ou "size"**

**Une balise de fin dont le nom de balise est "br", "p"**

[Erreur d'analyse](#) .

Tant que le [nœud actuel](#) n'est pas un [point d'intégration de texte MathML](#) , un [point d'intégration HTML](#) ou un élément de l' [espace de noms HTML](#) , extrayez les éléments de la [pile d'éléments ouverts](#) .

Retraitez le jeton selon les règles données dans la section correspondant au [mode d'insertion](#) courant dans le contenu HTML.

**Toute autre balise de début**

Si le [nœud actuel ajusté](#) est un élément de l' [espace de noms MathML](#) , [ajustez les attributs MathML](#) pour le jeton. (Cela corrige la casse des attributs MathML qui ne sont pas tous en minuscules.)

Si le [nœud actuel ajusté](#) est un élément de l' [espace de noms SVG](#) et que le nom de la balise du jeton est l'un de ceux de la première colonne du tableau suivant, remplacez le nom de la balise par le nom indiqué dans la cellule



correspondante de la deuxième colonne. (Cela corrige la casse des éléments SVG qui ne sont pas tous en minuscules.)

Nom de la balise	Nom de l'élément
altglyph	altGlyph
altglyphdef	altGlyphDef
altglyphitem	altGlyphItem
animatecolor	animateColor
animatemotion	animateMotion
animatetransform	animateTransform
clippath	clipPath
feblend	feBlend
fecolormatrix	feColorMatrix
fecomponenttransfer	feComponentTransfer
fecomposite	feComposite
feconvolvematrix	feConvolveMatrix
fediffuselighting	feDiffuseLighting
fedisplacementmap	feDisplacementMap
fedistantlight	feDistantLight
fedropshadow	feDropShadow
feflood	feFlood
fefunca	feFuncA
fefuncb	feFuncB
fefuncg	feFuncG
fefuncr	feFuncR
fegaussianblur	feGaussianBlur
feimage	feImage
femerge	feMerge
femergenode	feMergeNode
femorphology	feMorphology
feoffset	feOffset
fepointlight	fePointLight
fespecularlighting	feSpecularLighting
fespotlight	feSpotLight
fetile	feTile
feturbulence	feTurbulence
foreignobject	foreignObject
glyphref	glyphRef
lineargradient	linearGradient
radialgradient	radialGradient
textpath	textPath

Si le [nœud actuel ajusté](#) est un élément de l' [espace de noms SVG](#) , [ajustez les attributs SVG](#) pour le jeton. (Cela corrige la casse des attributs SVG qui ne sont pas tous en minuscules.)

[Ajustez les attributs étrangers](#) pour le jeton. (Cela corrige l'utilisation des attributs d'espace de noms, en particulier XLink dans SVG.)

[Insérez un élément étranger](#) pour le jeton, dans le même espace de noms que le [nœud actuel ajusté](#) .

Si le jeton a son [indicateur de fermeture automatique](#) défini, exécutez les étapes appropriées dans la liste suivante :

**Si le nom de la balise du jeton est "script" et que le nouveau [nœud actuel](#) se trouve dans l' [espace de noms SVG](#)**  
[Reconnaissez l' indicateur de fermeture automatique du jeton](#) , puis agissez comme décrit dans les étapes pour une balise de fin de "script" ci-dessous.

#### **Sinon**

Retirez le [nœud actuel](#) de la [pile d'éléments ouverts](#) et reconnaissez l' [indicateur de fermeture automatique du jeton](#) .

**Une balise de fin dont le nom de balise est "script", si le [nœud actuel](#) est un élément [SVG](#)[script](#)**

Détachez le [nœud actuel](#) de la [pile d'éléments ouverts](#) .

Laissez l' *ancien point d'insertion* avoir la même valeur que le [point d'insertion](#) actuel . Laissez le [point d'insertion](#) se trouver juste avant le [caractère d'entrée suivant](#) .

Incrémentez de un le [niveau d'imbrication des scripts de l'analyseur](#). Définissez l' [indicateur de pause de l'analyseur](#) sur vrai.

Si l' [analyseur HTML spéculatif actif](#) est nul et que l'agent utilisateur prend en charge SVG, [traitez l' \[script\]\(#\)élément](#) SVG conformément aux règles SVG. [\[SVG\]](#)

*Même si cela entraîne [l'insertion de nouveaux caractères dans le tokenizer](#) , l'analyseur ne sera pas exécuté de manière réentrante, car l' [indicateur de pause de l'analyseur](#) est vrai.*

Décrémentez le [niveau d'imbrication des scripts](#) de l'analyseur d'une unité. Si le [niveau d'imbrication du script](#) de l'analyseur est zéro, définissez l' [indicateur de pause de l'analyseur](#) sur faux.

Laissez au [point d'insertion](#) la valeur de l' *ancien point d'insertion* . (En d'autres termes, restaurez le [point d'insertion](#) à sa valeur précédente. Cette valeur peut être la valeur "indéfinie".)

**Toute autre balise de fin**

Exécutez ces étapes :

1. Initialiser *le nœud* pour qu'il soit le [nœud actuel](#) (le nœud le plus bas de la pile).
2. Si le nom de balise du *nœud* , [converti en minuscules ASCII](#) , n'est pas le même que le nom de balise du jeton, il s'agit d'une [erreur d'analyse](#) .
3. *Boucle* : Si *le nœud* est l'élément le plus haut dans la [pile d'éléments ouverts](#) , alors retour. ( [cas de fragment](#) )
4. Si le nom de balise de *node* , [converti en ASCII minuscule](#) , est le même que le nom de balise du jeton, extrayez les éléments de la [pile d'éléments ouverts](#) jusqu'à ce que *node* ait été extrait de la pile, puis revenez.
5. Définit *node* sur l'entrée précédente dans la [pile d'éléments ouverts](#) .
6. Si *node* n'est pas un élément dans l' [espace de noms HTML](#) , retournez à l'étape étiquetée *loop* .
7. Sinon, traitez le jeton selon les règles données dans la section correspondant au [mode d'insertion](#) courant dans le contenu HTML.

### 13.2.7 La fin



Une fois que l'agent utilisateur **arrête d'analyser** le document, l'agent utilisateur doit exécuter les étapes suivantes :



1. Si l' [analyseur HTML spéculatif actif](#) n'est pas nul, [arrêtez l'analyseur HTML spéculatif](#) et revenez.
2. Définissez le [point d'insertion](#) sur undefined.
3. [Mettre à jour la préparation actuelle du document](#) sur " *interactive*".
4. Détachez *tous* les nœuds de la [pile d'éléments ouverts](#) .
5. Alors que la [liste des scripts qui s'exécuteront lorsque le document aura terminé l'analyse](#) n'est pas vide :
  1. [Faites tourner la boucle d'événements](#) jusqu'à ce que le premier [script](#) dans la [liste des scripts qui s'exécuteront lorsque le](#)

document aura terminé l'analyse aura son prêt à être exécuté par l'analyseur défini sur `true` et l'analyseur `Document` n'a pas de feuille de style qui bloque les scripts .

2. Exécute l'élément de script donné par le premier `script` dans la liste des scripts qui s'exécuteront lorsque le document aura fini d'analyser .
3. Supprimez le premier `script` élément de la liste des scripts qui s'exécuteront lorsque le document aura terminé l'analyse (c'est-à-dire décaler la première entrée de la liste).
6. Mettez une tâche globale en file d'attente sur la source de la tâche de manipulation DOM en fonction de l' `Document` objet global pertinent de pour exécuter les sous-étapes suivantes :
  1. Définissez l' heure de début `Document` de l' événement chargé du contenu DOM des informations de synchronisation de chargement de sur l' heure haute résolution actuelle en fonction de l' objet global pertinent de `.Document`
  2. Déclenche un événement nommé `DOMContentLoaded` sur l' `Document` objet, avec son `bubbles` attribut initialisé à `true`.
  3. Définissez l' `Document` heure de fin de l' événement chargé du contenu DOM des informations de synchronisation de chargement de sur l' heure haute résolution actuelle en fonction de l' objet global pertinent de `.Document`
  4. Activez la file d'attente des messages du client de l' objet dont le client de service worker `ServiceWorkerContainer` associé est l' objet de paramètres pertinent de l'objet `.Document`
  5. Appelez le contenu DOM WebDriver BiDi chargé avec le contexte de navigation `Document` de et un nouveau statut de navigation WebDriver BiDi dont l'`id` est l' identifiant de navigation de l'objet , le statut est " " et l'`url` est l' URL de l'objet `.Document.pendingDocument`
7. Faites tourner la boucle d'événements jusqu'à ce que l' ensemble de scripts qui s'exécuteront dès que possible et la liste des scripts qui s'exécuteront dans l'ordre dès que possible soient vides.
8. Faites tourner la boucle d'événements jusqu'à ce qu'il n'y ait plus rien qui **retarde l'événement load** dans le fichier `Document`.
9. Mettez en file d'attente une tâche globale sur la source de la tâche de manipulation DOM en fonction de l' `Document` objet global pertinent de pour exécuter les étapes suivantes :
  1. Mettre à jour la préparation actuelle du document sur " `complete`".

2. Si le [contexte de navigation](#)[Document](#) de l'objet est nul, abandonnez ces étapes.
3. Soit *window* l' [objet global pertinent](#)[Document](#) de .
4. Définissez l' [heure de début de l' événement](#)[Document](#) de chargement de l' [information de synchronisation](#) de chargement sur la [fenêtre de temps haute résolution actuelle](#) donnée .
5. [Lancez un événement](#) nommé [load](#) à *window* , avec l'indicateur de *remplacement de cible hérité* défini.
6. Appelez [WebDriver BiDi load complet](#) avec le [Document](#) contexte [de navigation](#) de et un nouveau [statut de navigation WebDriver BiDi](#) dont l'[id](#) est l' [id de navigation](#)[Document](#) de l'objet , [le statut](#) est " " et l'[url](#) est l' [URL](#) de l'objet [.completeDocument](#)
7. Définissez l' [ID de navigation](#)[Document](#) de l'objet sur null.
8. Définissez l' [heure de fin de l' événement de chargement de l' information de synchronisation de chargement](#)[Document](#) sur la [fenêtre](#) de temps [haute résolution actuelle](#) donnée .
9. [Assert](#) : l' [affichage](#)[Document](#) de la page est faux.
10. Définissez l' indicateur d'affichage [Document](#) de la [page](#) sur true.
11. [Déclenchez un événement de transition de page](#) nommé [pageshow](#) à la *fenêtre* avec false.
12. [Terminez complètement le chargement](#) du fichier [Document](#).
13. [Mettre en file d'attente l'entrée de synchronisation de navigation](#) pour le [Document](#).

10. Si l' indicateur [Document](#) d' [impression lors du chargement](#) est activé, exécutez les [étapes d'impression](#) .

11. Le [Document](#) est maintenant **prêt pour les tâches de post-chargement** .

Lorsque l'agent utilisateur doit **abandonner un analyseur** , il doit exécuter les étapes suivantes :

1. Supprimez tout contenu en attente dans le [flux d'entrée](#) et supprimez tout contenu futur qui y aurait été ajouté.
2. [Arrêtez l'analyseur HTML spéculatif](#) pour cet analyseur HTML.
3. [Mettre à jour la préparation actuelle du document](#) sur " *interactive*".

4. Détachez *tous* les nœuds de la [pile d'éléments ouverts](#) .
5. [Mettre à jour la préparation actuelle du document](#) sur " complete".

### 13.2.8 Analyse HTML spéculative

Les agents utilisateurs peuvent implémenter une optimisation, comme décrit dans cette section, pour récupérer de manière spéculative les ressources qui sont déclarées dans le balisage HTML pendant que l'analyseur HTML attend qu'un script de blocage d'analyse en attente soit extrait et exécuté, [ou](#) pendant l'analyse normale, au moment où [un élément est créé pour un jeton](#) . Bien que cette optimisation ne soit pas définie avec précision, certaines règles doivent être prises en compte pour l'interopérabilité.

Chaque [analyseur HTML](#) peut avoir un **analyseur HTML spéculatif actif** . Il est initialement nul.

L' **analyseur HTML spéculatif** doit agir comme l'analyseur HTML normal (par exemple, les règles du générateur d'arbres s'appliquent), à quelques exceptions près :

- L'état de l'analyseur HTML normal et le document lui-même ne doivent pas être affectés.

Par exemple, le [caractère d'entrée suivant](#) ou la [pile d'éléments ouverts](#) pour l'analyseur HTML normal n'est pas affecté par l' [analyseur HTML spéculatif](#) .

- Les octets poussés dans le [flux d'octets d'entrée](#) de l'analyseur HTML doivent également être poussés dans le [flux d'octets d'entrée](#) de l'analyseur HTML spéculatif . Les octets lus à partir des flux doivent être indépendants.
- Le résultat de l'analyse spéculative est principalement une série de [récupérations spéculatives](#) . Les types de ressources à récupérer de manière spéculative sont [définis par l'implémentation](#) , mais les agents utilisateurs ne doivent pas récupérer de manière spéculative des ressources qui ne seraient pas récupérées avec l'analyseur HTML normal, sous l'hypothèse que le script qui bloque l'analyseur HTML ne fait rien.

*Il est possible que le même balisage soit vu plusieurs fois à partir de l' [analyseur HTML spéculatif](#) , puis de l'analyseur HTML normal. On s'attend à ce que les récupérations en double soient empêchées par des règles de mise en cache, qui ne sont pas encore entièrement spécifiées.*

Une **récupération spéculative** d'un élément [d'élément factice spéculatif](#) doit suivre ces règles :

Certaines de ces choses devraient-elles être appliquées au document "pour de vrai", même si elles sont trouvées de manière spéculative ?

- Si l' [analyseur HTML spéculatif](#) rencontre l'un des éléments suivants, agissez comme si cet élément était traité dans le but de son effet sur les récupérations spéculatives ultérieures.
  - Un [base](#)élément.
  - Un [meta](#)élément dont [http-equiv](#) l'attribut est dans l' état [de la politique de sécurité du contenu](#) .
  - Un [meta](#)élément dont [name](#) l'attribut est une correspondance [ASCII non sensible à la casse](#) pour "[referrer](#)".
  - Un [meta](#)élément dont [name](#) l'attribut est une correspondance [ASCII non sensible à la casse](#) pour "[viewport](#)". (Cela peut affecter si une liste de requêtes multimédias [correspond à l'environnement](#) .) [\[CSSDEVICEADAPT\]](#)
- Soit [url](#) l' [URL](#) que cet *élément* récupérerait s'il était traité normalement. S'il n'y a pas [d'URL](#) de ce type ou s'il s'agit d'une chaîne vide, ne faites rien. Sinon, si [url](#) figure déjà dans la [liste des URL de récupération spéculatives](#) , ne faites rien. Sinon, récupérez [url](#) comme si l'élément était traité normalement et ajoutez [url](#) à la [liste des URL de récupération spéculatives](#) .

Chacun [Document](#) a une **liste d'URL de récupération spéculative** , qui est une [liste](#) d' [URL](#) , initialement vide.

Pour **démarrer l'analyseur HTML spéculatif** pour une instance d'un analyseur *HTML* :

1. En option, retour.

*Cette étape permet aux agents utilisateurs de désactiver l'analyse HTML spéculative.*

2. Si l' *analyseur* [HTML spéculatif actif de parser](#) n'est pas nul, alors [arrêtez l'analyseur HTML spéculatif](#) pour *parser* .

*Cela peut se produire lors [document.write\(\)](#) de l'écriture d'un autre script bloquant l'analyseur. Pour plus de simplicité, cette spécification redémarre toujours l'analyse spéculative, mais les agents utilisateurs peuvent implémenter une stratégie plus efficace, tant que le résultat final est équivalent.*

3. Soit *speculativeParser* un nouvel [analyseur HTML spéculatif](#) , avec le même état que *parser* .
4. Soit *speculativeDoc* une nouvelle représentation isomorphe des *parseurs* , [Document](#) où tous les éléments sont à la place [des faux](#)

[éléments spéculatifs](#) . Laissez *speculativeParser* analyser dans *speculativeDoc* .

5. Définissez l'analyseur [HTML spéculatif actif](#) de l'analyseur sur *speculativeParser* .
6. [En parallèle](#) , exécutez *speculativeParser* jusqu'à ce qu'il soit arrêté ou jusqu'à ce qu'il atteigne la fin de son [flux d'entrée](#) .

Pour **arrêter le parseur HTML spéculatif** pour une instance d'un parseur *HTML* :

1. Soit *speculativeParser* l' analyseur HTML spéculatif [actif de l'analyseur](#) .
2. Si *speculativeParser* est null, alors retournez.
3. Supprimez tout contenu en attente dans [le flux d'entrée](#) de *speculativeParser* et supprimez tout contenu futur qui y aurait été ajouté.
4. Définissez [l'analyseur HTML spéculatif actif](#) de *parser* sur null.

L' [analyseur HTML spéculatif](#) créera [des éléments factices spéculatifs](#) au lieu d'éléments normaux. Les opérations DOM que le constructeur d'arbres effectue normalement sur les éléments doivent fonctionner correctement sur les éléments factices spéculatifs.

Un **élément factice spéculatif** est une [structure](#) avec les [éléments](#) suivants :

- Un **espace de noms** [de chaîne](#) , correspondant à [l'espace de noms](#) d'un élément .
- Un **nom local** [de chaîne](#) , correspondant au [nom local](#) d'un élément .
- Une **liste d'attributs** [list](#) , correspondant à la [liste d'attributs](#) d'un élément .
- Une [liste children](#) , correspondant aux [enfants](#) d'un élément .

Pour **créer un élément factice spéculatif** avec un *namespace* , un *tagName* et des *attributs* :

1. Soit *element* un nouvel [élément factice spéculatif](#) .
2. Définissez [l'espace de noms](#) de l' élément sur *namespace* .
3. Définissez [le nom local](#) de l'élément sur *tagName* .
4. Définissez [la liste d'attributs](#) de l'élément sur *attributs* .
5. Définissez [les enfants](#) de l'élément sur une nouvelle [liste](#) vide .
6. Effectuez éventuellement une [récupération spéculative](#) pour l'élément .



## 7. Élément de retour .

Lorsque le constructeur d'arbres dit d'insérer un élément dans le [contenu du modèle](#)`template` d'un élément , s'il s'agit d'un [élément factice spéculatif](#) , ne faites rien à la place. Les URL trouvées de manière spéculative à l'intérieur des éléments peuvent elles-mêmes être des modèles et ne doivent pas être extraites de manière spéculative.`template`

### 13.2.9 Contrainte d'un DOM HTML dans un ensemble d'informations

Lorsqu'une application utilise un [analyseur HTML](#) en conjonction avec un pipeline XML, il est possible que le DOM construit ne soit pas compatible avec la chaîne d'outils XML de certaines manières subtiles. Par exemple, une chaîne d'outils XML peut ne pas être en mesure de représenter des attributs avec le nom `xmlns`, car ils entrent en conflit avec les espaces de noms dans la syntaxe XML. Il existe également des données générées par l' [analyseur HTML](#) qui ne sont pas incluses dans le DOM lui-même. Cette section spécifie quelques règles pour gérer ces problèmes.

Si l'API XML utilisée ne prend pas en charge les DOCTYPE, l'outil peut supprimer complètement les DOCTYPE.

Si l'API XML ne prend pas en charge les attributs dans aucun espace de noms qui sont nommés " `xmlns`", les attributs dont les noms commencent par " `xmlns:`" ou les attributs dans l' [espace de noms XMLNS](#) , l'outil peut supprimer ces attributs.

L'outil peut annoter la sortie avec toutes les déclarations d'espace de noms requises pour un fonctionnement correct.

Si l'API XML utilisée limite les caractères autorisés dans les noms locaux des éléments et des attributs, l'outil peut mapper tous les noms locaux d'éléments et d'attributs que l'API ne prendrait pas en charge vers un ensemble de noms autorisés, en remplaçant n'importe *quel* caractère qui n'est pas pris en charge avec la lettre majuscule U et les six chiffres du point de code du caractère lorsqu'ils sont exprimés en hexadécimal, en utilisant les chiffres 0-9 et les lettres majuscules AF comme symboles, dans l'ordre numérique croissant.

Par exemple, le nom de l'élément `foo<bar`, qui peut être généré par l' [analyseur HTML](#) , bien qu'il ne s'agisse ni d'un nom d'élément HTML légal ni d'un nom d'élément XML bien formé, serait converti en `fooU000003Cbar`, qui est un nom d'élément XML bien formé (bien que ce n'est toujours pas légal en HTML par tous les moyens).

Comme autre exemple, considérons l'attribut `xlink:href`. Utilisé sur un élément MathML, il devient, après ajustement , un attribut avec un préfixe " `xlink`" et un nom local " `href`". Cependant, utilisé sur un élément HTML, il devient un attribut sans préfixe et le nom local " `xlink:href`", qui n'est pas un NCName valide, et peut donc

ne pas être accepté par une API XML. Il pourrait ainsi se convertir, devenir  
" xlinkU00003Ahref".

*Les noms résultant de cette conversion ne peuvent commodément entrer en conflit avec aucun attribut généré par l' [analyseur HTML](#) , car ceux-ci sont tous en minuscules ou ceux répertoriés dans la table de l'algorithme [d'ajustement des attributs étrangers](#) .*

Si l'API XML empêche les commentaires d'avoir deux caractères consécutifs U+002D HYPHEN-MINUS (--), l'outil peut insérer un seul caractère U+0020 SPACE entre ces caractères incriminés.

Si l'API XML empêche les commentaires de se terminer par un caractère U+002D HYPHEN-MINUS (-), l'outil peut insérer un seul caractère U+0020 SPACE à la fin de ces commentaires.

Si l'API XML restreint les caractères autorisés dans les données de caractères, les valeurs d'attribut ou les commentaires, l'outil peut remplacer tout caractère U+000C FORM FEED (FF) par un caractère U+0020 SPACE et tout autre caractère littéral non XML par un U +PERSONNAGE DE REMPLACEMENT FFFD.

Si l'outil n'a aucun moyen de transmettre des informations hors bande, il peut supprimer les informations suivantes :

- Si le document est défini sur [le mode sans bizarreries](#) , [le mode bizarreries limitées](#) ou [le mode bizarreries](#)
- L'association entre les contrôles de formulaire et les formulaires qui ne sont pas leur [form](#) ancêtre d'élément le plus proche (utilisation du [form](#)[pointeur d'élément](#) dans l'analyseur)
- Le [contenu du modèle](#) de tous [template](#) les éléments.

*Les mutations autorisées par cette section s'appliquent après l' application des règles de [l'analyseur HTML](#) . Par exemple, une <a :> balise de début sera fermée par une </a :> balise de fin, et jamais par une </aU00003AU00003A> balise de fin, même si l'agent utilisateur utilise les règles ci-dessus pour générer ensuite un élément réel dans le DOM avec le nom aU00003AU00003A de cette balise de début.*

### 13.2.10 Une introduction à la gestion des erreurs et des cas étranges dans l'analyseur

*Cette section est non normative.*

Cette section examine certains balisages erronés et explique comment l' [analyseur HTML](#) gère ces cas.

### 13.2.10.1 Balises mal imbriquées : <b><i></b></i>

Cette section est non normative.

L'exemple le plus souvent discuté de balisage erroné est le suivant :

```
<p>1<b>2<i>3</b>4</i>5</p>
```

L'analyse de ce balisage est simple jusqu'au "3". À ce stade, le DOM ressemble à ceci :

- [html](#)
  - [head](#)
  - [body](#)
    - [p](#)
      - [#text: 1](#)
      - [b](#)
        - [#text: 2](#)
        - [i](#)
          - [#text: 3](#)

Ici, la [pile d'éléments ouverts](#) contient cinq éléments : [html](#), [body](#), [p](#), [b](#) et [i](#). La [liste des éléments de formatage actifs](#) n'en contient que deux : [b](#) et [i](#). Le [mode d'insertion](#) est "dans le corps".

Lors de la réception du jeton de balise de fin avec le nom de balise "b", l'[algorithme de l'agence d'adoption](#) est invoqué. Il s'agit d'un cas simple, dans la mesure où l'*élément de mise en forme* est l'[b](#)élément et qu'il n'y a pas *de bloc le plus éloigné*. Ainsi, la [pile d'éléments ouverts](#) se retrouve avec seulement trois éléments : [html](#), [body](#), et [p](#), tandis que la [liste des éléments de formatage actifs](#) n'en contient qu'un : [i](#). L'arborescence DOM n'est pas modifiée à ce stade.

Le jeton suivant est un caractère ("4"), déclenche la [reconstruction des éléments de formatage actifs](#), dans ce cas uniquement l'[i](#)élément. Un nouvel [i](#)élément est ainsi créé pour le [Text](#)nœud "4". Une fois que le jeton de balise de fin pour le "i" est également reçu et que le [Text](#)nœud "5" est inséré, le DOM se présente comme suit :

- [html](#)
  - [head](#)
  - [body](#)
    - [p](#)
      - [#text: 1](#)
      - [b](#)
        - [#text: 2](#)
        - [i](#)
          - [#text: 3](#)
      - [i](#)
        - [#text: 4](#)
      - [#text: 5](#)

### 13.2.10.2 Balises mal imbriquées : <b><p></b></p>

Cette section est non normative.

Un cas similaire au précédent est le suivant :

```
<b>1<p>2</b>3</p>
```

Jusqu'au "2", l'analyse ici est simple :

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
      - [#text: 1](#)
      - [p](#)
        - [#text: 2](#)

La partie intéressante est lorsque le jeton de balise de fin avec le nom de balise "b" est analysé.

Avant que ce jeton ne soit vu, la [pile d'éléments ouverts](#) contient quatre éléments : [html](#), [body](#), [b](#) et [p](#). La [liste des éléments de formatage actifs](#) n'a qu'un seul : [b](#). Le [mode d'insertion](#) est " [dans le corps](#) ".

Lors de la réception du jeton de balise de fin avec le nom de balise "b", "l' [algorithme de l'agence d'adoption](#) " est invoqué, comme dans l'exemple précédent. Cependant, dans ce cas, il existe un *bloc le plus éloigné* , à savoir l' [p](#) élément. Ainsi, cette fois, l'algorithme de l'agence d'adoption n'est pas ignoré.

L' *ancêtre commun* est l' [body](#) élément. Un « signet » conceptuel marque la position du [b](#) dans la [liste des éléments de formatage actifs](#) , mais comme cette liste ne contient qu'un seul élément, le signet n'aura pas beaucoup d'effet.

Au fur et à mesure que l'algorithme progresse, *le nœud* finit par être défini sur l'élément de formatage ( [b](#) ), et *le dernier nœud* finit par être défini sur le *bloc le plus éloigné* ( [p](#) ).

Le *dernier nœud* est ajouté (déplacé) à l' *ancêtre commun* , de sorte que le DOM ressemble à :

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
      - [#text: 1](#)
    - [p](#)
      - [#text: 2](#)

Un nouvel [b](#) élément est créé et les enfants de l' [p](#) élément y sont déplacés :

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
    - [p](#)
      - [#text: 1](#)
- [b](#)
  - [#text: 2](#)

Enfin, le nouvel [b](#)élément est ajouté à l'[p](#)élément, de sorte que le DOM ressemble à :

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
    - [p](#)
      - [#text: 1](#)
      - [b](#)
        - [#text: 2](#)

L'[b](#)élément est supprimé de la [liste des éléments de formatage actifs](#) et de la [pile des éléments ouverts](#) , de sorte que lorsque le "3" est analysé, il est ajouté à l'[p](#)élément :

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
    - [p](#)
      - [#text: 1](#)
      - [b](#)
        - [#text: 2](#)
        - [#text: 3](#)

### 13.2.10.3 Balisage inattendu dans les tableaux

*Cette section est non normative.*

La gestion des erreurs dans les tables est, pour des raisons historiques, particulièrement étrange. Par exemple, considérez le balisage suivant :

```
<table><b><tr><td>aaa</td></tr>bbb</table>ccc
```

La [b](#)balise de début d'élément en surbrillance n'est pas autorisée directement à l'intérieur d'un tableau comme celui-ci, et l'analyseur gère ce cas en plaçant l'élément *avant* le tableau. (C'est ce qu'on appelle [le parentage adoptif](#) .) Cela peut être vu en examinant l'arborescence DOM telle qu'elle se présente juste après que la [table](#)balise de début de l'élément ait été vue :

- [html](#)
  - [head](#)
  - [body](#)
    - [table](#)

... puis immédiatement après que la [b](#) balise de début de l'élément ait été vue :

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
    - [table](#)

À ce stade, la [pile d'éléments ouverts](#) contient les éléments [html](#), [body](#), [table](#), et [b](#) (dans cet ordre, malgré l'arbre DOM résultant) ; la [liste des éléments de formatage actifs](#) ne contient que l' [b](#) élément ; et le [mode d'insertion](#) est " [dans le tableau](#) ".

La [tr](#) balise de début entraîne le [b](#) retrait de l'élément de la pile et [tbody](#) l'implication d'une balise de début ; les éléments [tbody](#) et [tr](#) sont ensuite traités de manière assez simple, en faisant passer l'analyseur par les modes d'insertion " [dans le corps de la table](#) " et " [dans la ligne](#) ", après quoi le DOM se présente comme suit :

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
    - [table](#)
      - [tbody](#)
        - [tr](#)

Ici, la [pile d'éléments ouverts](#) contient les éléments [html](#), [body](#), [table](#), [tbody](#), et [tr](#) ; la [liste des éléments de formatage actifs](#) contient toujours l' [b](#) élément ; et le [mode d'insertion](#) est " [en ligne](#) ".

Le [td](#) jeton de balise de début d'élément, après avoir placé un [td](#) élément sur l'arbre, place un [marqueur](#) sur la [liste des éléments de mise en forme actifs](#) (il passe également en [mode d'insertion " dans la cellule "](#)).

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
    - [table](#)
      - [tbody](#)
        - [tr](#)
          - [td](#)

Le [marqueur](#) signifie que lorsque les jetons de caractère "aaa" sont vus, aucun élément n'est créé pour contenir le nœud [b](#) résultant : [Text](#)

- [html](#)

- head
- body
  - b
  - table
    - tbody
      - tr
        - td
          - #text: aaa

Les balises de fin sont gérées de manière simple ; après les avoir manipulés, la pile d'éléments ouverts contient les éléments html, body, table, et tbody; la liste des éléments de formatage actifs contient toujours l' bélément (le marqueur ayant été supprimé par le jeton de balise de fin "td") ; et le mode d'insertion est " dans le corps du tableau ".

C'est ainsi que les jetons de caractère "bbb" sont trouvés. Ceux-ci déclenchent l' utilisation du mode d'insertion " dans le texte du tableau " (avec le mode d'insertion d'origine défini sur " dans le corps du tableau "). Les jetons de personnage sont collectés, et lorsque le jeton suivant (la table balise de fin d'élément) est vu, ils sont traités comme un groupe. Comme ce ne sont pas tous des espaces, ils sont traités selon les règles "tout le reste" dans le mode d'insertion " dans le tableau ", qui s'en remettent au mode d'insertion " dans le corps " mais avec le parentage adoptif .

Lorsque les éléments de mise en forme actifs sont reconstruits , un bélément est créé et adopte un parent nourricier , puis le Text nœud "bbb" lui est ajouté :

- html
  - head
  - body
    - b
    - b
      - #text: bbb
    - table
      - tbody
        - tr
          - td
            - #text: aaa

La pile d'éléments ouverts contient les éléments html, body, table, tbody, et le new b(encore une fois, notez que cela ne correspond pas à l'arbre résultant !) ; la liste des éléments de formatage actifs contient le nouvel bélément ; et le mode d'insertion est toujours " dans le corps du tableau ".

Si les jetons de caractère n'avaient été que des espaces blancs ASCII au lieu de "bbb", alors cet espace blanc ASCII aurait simplement été ajouté à l' tbody élément.

Enfin, le table est fermé par une balise de fin "table". Cela fait apparaître tous les nœuds de la pile d'éléments ouverts jusqu'à et y compris l' table élément , mais cela n'affecte pas la liste des éléments de formatage actifs b , donc les jetons de caractère "ccc" après le tableau entraînent la création d'un autre élément, ceci temps après la table:

- [html](#)
  - [head](#)
  - [body](#)
    - [b](#)
    - [b](#)
      - [#text:](#) *bbb*
    - [table](#)
      - [tbody](#)
        - [tr](#)
          - [td](#)
            - [#text:](#) *aaa*
    - [b](#)
      - [#text:](#) *ccc*

#### 13.2.10.4 Scripts qui modifient la page lors de son analyse

*Cette section est non normative.*

Considérez le balisage suivant, qui pour cet exemple, nous supposons qu'il s'agit du document avec [URL](https://example.com/inner) `https://example.com/inner`, rendu comme le contenu d'un [iframe](#) dans un autre document avec l' [URL](https://example.com/outer) `https://example.com/outer` :

```
<div id=a>
  <script>
    var div = document.getElementById('a');
    parent.document.body.appendChild(div);
  </script>
  <script>
    alert(document.URL);
  </script>
</div>
<script>
  alert(document.URL);
</script>
```

Jusqu'à la première balise de fin "script", avant que le script ne soit analysé, le résultat est relativement simple :

- [html](#)
  - [head](#)
  - [body](#)
    - [div id=" a "](#)
      - [#text:](#)
      - [script](#)
        - [#text:](#) `var div = document.getElementById('a');` ↵  
`parent.document.body.appendChild(div);`



Une fois le script analysé, cependant, l' divélément et son scriptélément enfant ont disparu :

- html
  - head
  - body

Ils sont, à ce stade, dans le contexte de navigationDocument externe susmentionné . Cependant, la pile d'éléments ouverts *contient toujours l' élément . div*

Ainsi, lorsque le deuxième scriptélément est analysé, il est inséré *dans l' Documentobjet externe* .

Ceux analysés dans des s différents Documentde celui pour lequel l'analyseur a été créé ne s'exécutent pas, donc la première alerte ne s'affiche pas.

Une fois que la divbalise de fin de l'élément est analysée, l' divélément est retiré de la pile, et donc l' scriptélément suivant est dans l'inner Document:

- html
  - head
  - body
    - script
      - #text: *alerte(document.URL);*

Ce script s'exécute, entraînant une alerte indiquant "https://example.com/inner".

### 13.2.10.5 L'exécution de scripts qui se déplacent sur plusieurs documents

*Cette section est non normative.*

En élaborant sur l'exemple de la section précédente, considérons le cas où le deuxième scriptélément est un script externe (c'est-à-dire un script avec un srcattribut). Étant donné que l'élément n'était pas dans l'analyseur Documentlors de sa création, ce script externe n'est même pas téléchargé.

Dans le cas où un scriptélément avec un src attribut est analysé normalement dans son analyseur Document, mais pendant que le script externe est en cours de téléchargement, l'élément est déplacé vers un autre document, le script continue à se télécharger, mais ne s'exécute pas.

*En général, déplacer scriptdes éléments entre Documents est considéré comme une mauvaise pratique.*

### 13.2.10.6 Éléments de formatage non fermés

*Cette section est non normative.*

Le balisage suivant montre comment les éléments de mise en forme imbriqués (tels que **b**) sont collectés et continuent d'être appliqués même lorsque les éléments dans lesquels ils sont contenus sont fermés, mais que les doublons excessifs sont supprimés.

```
<!DOCTYPE html>
<p><b class=x><b class=x><b><b class=x><b class=x><b>X
<p>X
<p><b><b class=x><b>X
<p></b></b></b></b></b></b></b>X
```

L'arborescence DOM résultante est la suivante :

- DOCTYPE :html
  - [html](#)
    - [head](#)
    - [body](#)
      - [p](#)
        - [b class=" x"](#)
          - [b class=" x"](#)
            - [b](#)
              - [b class=" x"](#)
                - [b class=" x"](#)
                  - [b](#)
                    - [#text: X](#)
  - [p](#)
    - [b class=" x"](#)
      - [b](#)
        - [b class=" x"](#)
          - [b class=" x"](#)
            - [b](#)
              - [#text: X](#)
- [p](#)
  - [b class=" x"](#)
    - [b](#)
      - [b class=" x"](#)
        - [b class=" x"](#)
          - [b](#)
            - [b](#)
              - [b class=" x"](#)
                - [b](#)
                  - [#text: X](#)
- [p](#)
  - [#text: X](#)

Notez comment le deuxième pélément du balisage n'a pas b d'éléments explicites, mais dans le DOM résultant, jusqu'à trois de chaque type d'élément de mise en forme (dans ce cas, trois béléments avec l'attribut class et deux béléments sans ornement) sont reconstruits avant que l'élément " X".

Notez également comment cela signifie que dans le dernier paragraphe, seules six bbalises de fin sont nécessaires pour effacer complètement la [liste des éléments de formatage actifs](#) , même si neuf bbalises de début ont été vues jusqu'à présent.

### 13.3 Sérialisation de fragments HTML

Pour les besoins de l'algorithme suivant, un élément **est sérialisé en tant que void** si son type d'élément est l'un des [éléments void](#) , ou est [basefont](#), [bgsound](#), [frame](#) ou [keygen](#).

Les étapes suivantes forment l' **algorithme de sérialisation des fragments HTML** . L'algorithme prend en entrée un DOM [Element](#), [Document](#) ou [DocumentFragment](#) appelé *le nœud* , et renvoie une chaîne.

*Cet algorithme sérialise les enfants du nœud en cours de sérialisation, pas le nœud lui-même.*

1. Si *le nœud* [est sérialisé en tant que void](#) , renvoie la chaîne vide.
2. Soit *s* une chaîne et initialisons-la avec la chaîne vide.
3. Si *le nœud* est un [template](#)élément, laissez *le nœud* être à la place le [contenu du modèle](#)[template](#) de l'élément (un nœud).[DocumentFragment](#)
4. Pour chaque nœud enfant du *nœud* , dans [l'ordre de l'arborescence](#) , exécutez les étapes suivantes :
  1. Soit *le nœud actuel* le nœud enfant en cours de traitement.
  2. Ajoutez la chaîne appropriée de la liste suivante à *s* :

**Si *le nœud actuel* est un [Element](#)**

Si *le nœud actuel* est un élément de l' [espace de noms HTML](#) , de l' [espace de noms MathML](#) ou de l' [espace de noms SVG](#) , laissez *tagname* être le nom local du *nœud actuel* . Sinon, laissez *tagname* être le nom qualifié du *nœud actuel* .

Ajoutez un caractère SIGNE MOINS-QUE U+003C (<), suivi de *tagname* .

*Pour les éléments HTML créés par l' [analyseur HTML](#) ou [createElement\(\)](#) , le nom de balise sera en minuscules.*

Si [la valeur](#) du *nœud actuel* n'est pas nulle et que l'élément n'a pas d'attribut dans sa liste d'attributs, ajoutez la chaîne " ", suivie de [la valeur](#) du *nœud actuel* [échappée comme décrit ci-dessous](#) en *mode attribut* , suivi d'un U+0022 guillemet ("). `.isis is="is`

Pour chaque attribut de l'élément, ajoutez un caractère U+0020 ESPACE, le [nom sérialisé de l'attribut comme décrit ci-dessous](#) , un caractère U+003D SIGNE ÉGAL (=), un caractère U+0022 GUILLEMET ("), la valeur de l'attribut, [échappé comme décrit ci-dessous](#) en *mode attribut* , et un deuxième caractère U+0022 GUILLEMET (").

Le **nom sérialisé d'un attribut** aux fins du paragraphe précédent doit être déterminé comme suit :

**Si l'attribut n'a pas d'espace de noms**

Le nom sérialisé de l'attribut est le nom local de l'attribut.

*Pour les attributs sur [les éléments HTML](#) définis par l' [analyseur HTML](#) ou par `setAttribute()` , le nom local sera en minuscules.*

**Si l'attribut est dans l' [espace de noms XML](#)**

Le nom sérialisé de l'attribut est la chaîne " `xml:` " suivie du nom local de l'attribut.

**Si l'attribut se trouve dans l' [espace de noms XMLNS](#) et que le nom local de l'attribut est `xmlns`**

Le nom sérialisé de l'attribut est la chaîne " `xmlns` ".

**Si l'attribut est dans l' [espace de noms XMLNS](#) et que le nom local de l'attribut n'est pas `xmlns`**

Le nom sérialisé de l'attribut est la chaîne " `xmlns:` " suivie du nom local de l'attribut.

**Si l'attribut est dans l' [espace de noms XLink](#)**

Le nom sérialisé de l'attribut est la chaîne " `xlink:` " suivie du nom local de l'attribut.

**Si l'attribut est dans un autre espace de noms**

Le nom sérialisé de l'attribut est le nom qualifié de l'attribut.

Alors que l'ordre exact des attributs est [défini par l'implémentation](#) et peut dépendre de facteurs tels que l'ordre dans lequel les attributs ont été donnés dans le balisage d'origine, l'ordre de tri doit être stable, de sorte que les invocations consécutives de cet algorithme sérialisent les attributs d'un élément dans le même ordre.

Ajoutez un caractère SIGNE SUPÉRIEUR À U+003E (>).

Si le *nœud actuel* [est sérialisé en tant que void](#) , [passez](#) au nœud enfant suivant à ce stade.

Ajoutez la valeur d'exécution de l' [algorithme de sérialisation de fragment HTML](#) sur l' élément *de nœud actuel* (récursif dans cet algorithme pour cet élément), suivi d'un caractère U+003C LESS-THAN SIGN (<), d'un caractère U+002F SOLIDUS (/) , *tagname* à nouveau, et enfin un caractère SIGNE SUPÉRIEUR À U+003E (>).

**Si le nœud courant est un [Text](#)nœud**

Si le parent du *nœud actuel* est un élément [style](#), [script](#), [xmp](#), [iframe](#), [noembed](#), [noframes](#), ou [plaintext](#), ou si le parent du *nœud actuel* est un [noscript](#) élément et que [le script est activé](#) pour le nœud, alors ajoutez littéralement la valeur des [données](#) du *nœud actuel* .

Sinon, ajoutez la valeur des [données](#) du *nœud actuel* , [échappées comme décrit ci-dessous](#) .

**Si le nœud actuel est un [Comment](#)**

Ajoutez la chaîne littérale "<!--" (U+003C SIGNE MOINS-QUE, U+0021 POINT D'EXCLAMATION, U+002D Trait d'union-moins, U+002D Trait d'union-moins), suivi de la valeur des données du nœud [actuel](#) , *suivi* du chaîne littérale " " (U+002D Trait d'union-Moins, U+002D Trait d'union-Moins, U+003E SIGNE SUPÉRIEUR À).-->

**Si le nœud actuel est un [ProcessingInstruction](#)**

Ajoutez la chaîne littérale "<?" (U+003C SIGNE MOINS-QUE, U+003F POINT D'INTERROGATION), suivi de la valeur de l' attribut IDL du *nœud actuel*/*target* , suivi d'un seul caractère ESPACE U+0020, suivi de la valeur de l'attribut *actuel* [données](#) du *nœud* , suivies d'un seul caractère U+003E SIGNE SUPÉRIEUR À (>).

**Si le nœud actuel est un [DocumentType](#)**

Ajoutez la chaîne littérale "<!DOCTYPE" (U+003C SIGNE INFÉRIEUR À, U+0021 POINT D'EXCLAMATION, U+0044 LETTRE MAJUSCULE LATINE D, U+004F LETTRE MAJUSCULE LATINE O, U+0043 LETTRE MAJUSCULE LATINE C, U+0054 MAJUSCULE LATINE LETTRE T, U+0059 LETTRE MAJUSCULE LATINE Y, U+0050 LETTRE MAJUSCULE LATINE P, U+0045 LETTRE MAJUSCULE LATINE E), suivi d'un espace (U+0020 ESPACE), suivi de la valeur du nom [du](#) nœud *actuel* , suivi de la chaîne littérale " " (U+003E SIGNE SUPÉRIEUR À).>

5. Retour s .

**Il est possible que la sortie de cet algorithme, si elle est analysée avec un [analyseur HTML](#) , ne renvoie pas l'arborescence d'origine. Les structures arborescentes qui n'effectuent pas d'aller-retour entre une étape de sérialisation et d'analyse peuvent également être produites par l' [analyseur HTML](#) lui-même, bien que de tels cas soient généralement non conformes.**

Par exemple, si un [textarea](#) élément auquel un [Comment](#) nœud a été ajouté est sérialisé et que la sortie est ensuite analysée, le commentaire finira par s'afficher dans le contrôle de texte. De même, si, à la suite d'une manipulation DOM, un

élément contient un commentaire qui contient la chaîne littérale " -->", alors lorsque le résultat de la sérialisation de l'élément est analysé, le commentaire sera tronqué à ce point et le reste du commentaire sera être interprété comme un balisage. D'autres exemples seraient de faire en sorte qu'un `script` élément contienne un `Text` nœud avec la chaîne de texte " </script>", ou d'avoir un `p` élément qui contient un `ul` élément (car la balise de début`ul` de l'élément impliquerait la balise de fin pour le `p`).

Cela peut permettre des attaques de script intersite. Un exemple de ceci serait une page qui permet à l'utilisateur d'entrer des noms de famille de polices qui sont ensuite insérés dans un `style` bloc CSS via le DOM et qui utilise ensuite l' `innerHTML` attribut IDL pour obtenir la sérialisation HTML de cet `style` élément : si l'utilisateur entre " </style><script>attack</script>" en tant que nom de famille de polices, `innerHTML` renverra un balisage qui, s'il était analysé dans un contexte différent, contiendrait un `script` nœud, même si aucun `script` nœud n'existait dans le DOM d'origine.

Par exemple, considérez le balisage suivant :

```
<form id="outer"><div></form><form id="inner"><input>
```

Cela sera analysé en :

- `html`
  - `head`
  - `body`
    - `form id=" outer"`
      - `div`
        - `form id=" inner"`
          - `input`

L' `input` élément sera associé à l' `form` élément intérieur. Désormais, si cette structure arborescente est sérialisée et analysée, la `<form id="inner">` balise de début sera ignorée et l' `input` élément sera donc associé à l' `form` élément externe à la place.

```
<html><head></head><body><form id="outer"><div><form id="inner"><input></form></div></form></body></html>
```

- `html`
  - `head`
  - `body`
    - `form id=" outer"`
      - `div`
        - `input`

Comme autre exemple, considérez le balisage suivant :

```
<a><table><a>
```

Cela sera analysé en :

- `html`

- [head](#)
- [body](#)
  - [a](#)
    - [a](#)
    - [table](#)

C'est-à-dire que les [a](#)éléments sont imbriqués, car le deuxième [a](#)élément est [adoptif parenté](#) . Après un aller-retour sérialiser-réparer, les [a](#)éléments et l' [table](#)élément seraient tous frères, car la deuxième `<a>` balise de début ferme implicitement le premier [a](#) élément.

```
<html><head></head><body><a><a></a><table></table></a></body></html>
```

- [html](#)
  - [head](#)
  - [body](#)
    - [a](#)
    - [a](#)
    - [table](#)

Pour des raisons historiques, cet algorithme n'effectue pas d'aller-retour avec un caractère initial U+000A LINE FEED (LF) dans [pre](#) les éléments [textarea](#), ou , même si (dans les deux premiers cas) le balisage faisant l'objet d'un aller-retour peut être conforme. [listing](#) L' [analyseur HTML](#) supprimera un tel caractère lors de l'analyse, mais cet algorithme ne sérialise pas un caractère supplémentaire U + 000A LINE FEED (LF).

Par exemple, considérez le balisage suivant :

```
<pre>

Hello.</pre>
```

Lorsque ce document est analysé pour la première fois, le [contenu du texte enfant](#)[pre](#) de l'élément commence par un seul caractère de saut de ligne. Après un aller-retour de sérialisation-réanalyse, le [contenu du texte enfant](#) de l'élément est simplement " ".[pre](#)Hello.

En raison du rôle particulier de l' [is](#)attribut dans la signalisation de la création d' [éléments intégrés personnalisés](#) , en ce sens qu'il fournit un mécanisme permettant au code HTML analysé de définir la [is valeur](#) de l'élément , nous spécifions sa gestion lors de la sérialisation. [is](#)[Cela garantit que la valeur](#) d'un élément est préservée grâce aux allers-retours de sérialisation-analyse.

Lors de la création d'un [élément intégré personnalisé](#) via l'analyseur, un développeur utilise [is](#)directement l'attribut ; dans de tels cas, les allers-retours de sérialisation-analyse fonctionnent bien.

```
<script>

window.SuperP = class extends HTMLParagraphElement {};
```

```

customElements.define("super-p", SuperP, { extends: "p" });
</script>

<div id="container"><p is="super-p">Superb!</p></div>

<script>
console.log(container.innerHTML); // <p is="super-p">
container.innerHTML = container.innerHTML;
console.log(container.innerHTML); // <p is="super-p">
console.assert(container.firstChild instanceof SuperP);
</script>

```

Mais lors de la création d'un élément intégré personnalisé via son [constructeur](#) ou via [createElement\(\)](#), l'[is](#) attribut n'est pas ajouté. Au lieu de cela, la [is](#) valeur (qui est celle utilisée par la machinerie des éléments personnalisés) est définie sans intermédiaire via un attribut.

```

<script>
container.innerHTML = "";
const p = document.createElement("p", { is: "super-p" });
container.appendChild(p);

// The is attribute is not present in the DOM:
console.assert(!p.hasAttribute("is"));

// But the element is still a super-p:
console.assert(p instanceof SuperP);
</script>

```

Pour garantir que les allers-retours de sérialisation-analyse fonctionnent toujours, le processus de sérialisation écrit explicitement la [is](#) valeur de l'élément en tant [is](#) qu'attribut :

```

<script>
console.log(container.innerHTML); // <p is="super-p">
container.innerHTML = container.innerHTML;
console.log(container.innerHTML); // <p is="super-p">
console.assert(container.firstChild instanceof SuperP);
</script>

```

**L'échappement d'une chaîne** (pour les besoins de l'algorithme ci-dessus) consiste à exécuter les étapes suivantes :

1. Remplacez toute occurrence du caractère " " par la chaîne " & ".



2. Remplacez toutes les occurrences du caractère U+00A0 NO-BREAK SPACE par la chaîne " &nbsp;".
3. Si l'algorithme a été invoqué en *mode attribut*, remplacez toutes les occurrences du "caractère " " par la chaîne " &quot;".
4. Si l'algorithme *n'a pas* été appelé en *mode attribut*, remplacez toutes les occurrences du <caractère " " par la chaîne " &lt;", et toutes les occurrences du >caractère " " par la chaîne " &gt;".

## 13.4 Analyser des fragments HTML

Les étapes suivantes forment l' **algorithme d'analyse de fragment HTML**. L'algorithme prend en entrée un Element nœud, appelé élément **de contexte**, qui donne le contexte pour l'analyseur, ainsi que *input*, une chaîne à analyser, et renvoie une liste de zéro ou plusieurs nœuds.

*Les parties marquées en casse fragment dans les algorithmes de la section analyseur sont des parties qui n'apparaissent que si l'analyseur a été créé pour les besoins de cet algorithme. Les algorithmes ont été annotés avec de tels marquages à des fins d'information uniquement ; ces marquages n'ont aucun poids normatif. S'il est possible qu'une condition décrite comme un cas de fragment se produise même lorsque l'analyseur n'a pas été créé dans le but de gérer cet algorithme, il s'agit alors d'une erreur dans la spécification.*

1. Créez un nouveau Document nœud et marquez-le comme étant un document HTML.
2. Si le document de nœud de l'élément de contexte est en mode quirks, alors laissez-le Document être en mode quirks. Sinon, le document de nœud de l'élément de contexte est en mode limited-quirks, alors laissez-le Document être en mode limited-quirks. Sinon, laissez le Document en mode sans fioritures.
3. Créez un nouvel analyseur HTML et associez-le au Document nœud que vous venez de créer.
4. Définissez l'état de l'étape de tokenisation de l'analyseur HTML comme suit, en activant l'élément de contexte :

title  
textarea

Passez le tokenizer à l' état RCDATA.

style  
xmp  
iframe  
noembed  
noframes

Passez le tokenizer à l' [état RAWTEXT](#) .

script

Basculez le tokenizer vers l' [état des données de script](#) .

noscript

Si l' [indicateur de script](#) est activé, passez le tokenizer à l' [état RAWTEXT](#) . Sinon, laissez le tokenizer dans l' [état data](#) .

plaintext

Passez le tokenizer à l' [état PLAINTEXT](#) .

### **Tout autre élément**

Laissez le tokenizer dans l' [état data](#) .

*Pour des raisons de performances, une implémentation qui ne signale pas d'erreurs et qui utilise directement la machine d'état réelle décrite dans cette spécification pourrait utiliser l'état PLAINTEXT au lieu des états RAWTEXT et de données de script lorsque ceux-ci sont mentionnés dans la liste ci-dessus. À l'exception des règles concernant les erreurs d'analyse, elles sont équivalentes, car il n'y a pas de [jeton de balise de fin approprié](#) dans le cas du fragment, mais elles impliquent beaucoup moins de transitions d'état.*

5. Soit *root* un nouvel [html](#) élément sans attribut.
6. Ajoutez la *racine* de l'élément au [Document](#) nœud créé ci-dessus.
7. Configurez la [pile d'éléments ouverts](#) de l'analyseur afin qu'elle ne contienne qu'un seul élément *root* .
8. Si l' élément [de contexte](#) est un [template](#) élément, poussez " [dans le modèle](#) " sur la [pile des modes d'insertion de modèle](#) afin qu'il soit le nouveau [mode d'insertion de modèle actuel](#) .
9. Créez un jeton de balise de début dont le nom est le nom local de [context](#) et dont les attributs sont les attributs de [context](#) .

Laissez ce jeton de balise de début être le jeton de balise de début du nœud [de contexte](#) , par exemple pour déterminer s'il s'agit d'un [point d'intégration HTML](#) .

10. [Réinitialisez le mode d'insertion de l'analyseur de manière appropriée](#) .

*L'analyseur fera référence à l' élément [de contexte](#) dans le cadre de cet algorithme.*

11. Définissez le [form](#) [pointeur d'élément](#) de l'analyseur sur le nœud le plus proche de l' élément [de contexte](#) qui est un [form](#) élément (en remontant directement la chaîne d'ancêtres et en incluant l'élément lui-même, s'il s'agit d'un [form](#) élément), le cas échéant. (Si un tel élément n'existe pas [form](#), le [form](#) [pointeur d'élément](#) conserve sa valeur initiale, null.)

12. Placez l' *entrée* dans le [flux d'entrée](#) pour l' [analyseur HTML](#) que vous venez de créer. La [confiance](#) d'encodage n'est *pas pertinente* .
13. Démarrez l'analyseur et laissez-le s'exécuter jusqu'à ce qu'il ait consommé tous les caractères qui viennent d'être insérés dans le flux d'entrée.
14. Renvoie les nœuds enfants de *root* , dans [l'ordre de l'arborescence](#) .
- 15.

# 1. [13.5 Références de caractères nommés](#)

## 13.5 Références de caractères nommés

Ce tableau répertorie les noms de référence de caractères pris en charge par HTML et les points de code auxquels ils font référence. Il est référencé par les sections précédentes.

*Il est intentionnel, pour la compatibilité héritée, que de nombreux points de code aient plusieurs noms de référence de caractères. Par exemple, certains apparaissent à la fois avec et sans le point-virgule final, ou avec des majuscules différentes.*

| Nom     | Personnages)    | Glyphe         |
|---------|-----------------|----------------|
| Aacute; | U+000C1         | UN             |
| Aacute  | U+000C1         | UN             |
| aacute; | U+000E1         | un             |
| aacute  | U+000E1         | un             |
| Abreve; | U+00102         | UN             |
| abreve; | U+00103         | un             |
| ac;     | U+0223E         | ∞              |
| acd;    | U+0223F         | ~              |
| acE;    | U+0223E U+00333 | ∞ <sub>=</sub> |
| Acirc;  | U+000C2         | UN             |
| Acirc   | U+000C2         | UN             |
| acirc;  | U+000E2         | un             |
| acirc   | U+000E2         | un             |
| acute;  | U+000B4         | '              |
| acute   | U+000B4         | '              |
| Acy;    | U+00410         | À              |
| acy;    | U+00430         | un             |

| Nom       | Personnages) | Glyphe         |
|-----------|--------------|----------------|
| Ælig;     | U+000C6      | Æ              |
| Ælig      | U+000C6      | Æ              |
| ælig;     | U+000E6      | æ              |
| ælig      | U+000E6      | æ              |
| af;       | U+02061      | $\mathbb{f}_0$ |
| Afr;      | U+1D504      | ℒ              |
| afr;      | U+1D51E      | ɑ              |
| Agrave;   | U+000C0      | UN             |
| Agrave    | U+000C0      | UN             |
| agrave;   | U+000E0      | un             |
| agrave    | U+000E0      | un             |
| alefsym;  | U+02135      | ℵ              |
| aleph;    | U+02135      | ℵ              |
| Alpha;    | U+00391      | Α              |
| alpha;    | U+003B1      | α              |
| Amacr;    | U+00100      | UN             |
| amacr;    | U+00101      | un             |
| amalg;    | U+02A3F      | ℒ              |
| AMP;      | U+00026      | &              |
| AMP       | U+00026      | &              |
| amp;      | U+00026      | &              |
| amp       | U+00026      | &              |
| And;      | U+02A53      | ⋈              |
| and;      | U+02227      | ∧              |
| andand;   | U+02A55      | ⋈              |
| andd;     | U+02A5C      | ⋈              |
| andslope; | U+02A58      | ∧              |
| andv;     | U+02A5A      | ⋈              |
| ang;      | U+02220      | ∠              |
| ange;     | U+029A4      | ⋈              |
| angle;    | U+02220      | ∠              |
| angmsd;   | U+02221      | ⋈              |
| angmsdaa; | U+029A8      | ⋈              |
| angmsdab; | U+029A9      | ⋈              |
| angmsdac; | U+029AA      | ⋈              |
| angmsdad; | U+029AB      | ⋈              |

| Nom            | Personnages) | Glyphe        |
|----------------|--------------|---------------|
| angmsdae;      | U+029AC      | ↗             |
| angmsdaf;      | U+029AD      | ↘             |
| angmsdag;      | U+029AE      | ↗↘            |
| angmsdah;      | U+029AF      | ↗↘↗           |
| angrt;         | U+0221F      | └             |
| angrtvb;       | U+022BE      | └             |
| angrtvbd;      | U+0299D      | └             |
| angsph;        | U+02222      | ↗             |
| angst;         | U+000C5      | UN            |
| angzarr;       | U+0237C      | └             |
| Aogon;         | U+00104      | UN            |
| aogon;         | U+00105      | un            |
| Aopf;          | U+1D538      | ℳ             |
| aopf;          | U+1D552      | ℳ             |
| ap;            | U+02248      | ≈             |
| apacir;        | U+02A6F      | ≈             |
| apE;           | U+02A70      | ≅             |
| ape;           | U+0224A      | ≈             |
| apid;          | U+0224B      | ≈             |
| apos;          | U+00027      | '             |
| ApplyFunction; | U+02061      | $\mathcal{f}$ |
| approx;        | U+02248      | ≈             |
| approxeq;      | U+0224A      | ≈             |
| Aring;         | U+000C5      | UN            |
| Aring          | U+000C5      | UN            |
| aring;         | U+000E5      | un            |
| aring          | U+000E5      | un            |
| Ascr;          | U+1D49C      | ℳ             |
| ascr;          | U+1D4B6      | ℳ             |
| Assign;        | U+02254      | :=            |
| ast;           | U+0002A      | *             |
| asyp;          | U+02248      | ≈             |
| asympeq;       | U+0224D      | ≈             |
| Atilde;        | U+000C3      | UN            |
| Atilde         | U+000C3      | UN            |
| atilde;        | U+000E3      | un            |

| Nom          | Personnages) | Glyphe                  |
|--------------|--------------|-------------------------|
| atilde       | U+000E3      | un                      |
| Auml;        | U+000C4      | UN                      |
| Auml         | U+000C4      | UN                      |
| auml;        | U+000E4      | un                      |
| auml         | U+000E4      | un                      |
| awconint;    | U+02233      | $\oint$                 |
| awint;       | U+02A11      | $\int$                  |
| backcong;    | U+0224C      | $\cong$                 |
| backepsilon; | U+003F6      | $\epsilon$              |
| backprime;   | U+02035      | $\text{'}^{\backslash}$ |
| backsim;     | U+0223D      | $\sim$                  |
| backsimeq;   | U+022CD      | $\simeq$                |
| Backslash;   | U+02216      | $\backslash$            |
| Barv;        | U+02AE7      | $\varepsilon$           |
| barvee;      | U+022BD      | $\bar{\vee}$            |
| Barwed;      | U+02306      | $\bar{\wedge}$          |
| barwed;      | U+02305      | $\bar{\wedge}$          |
| barwedge;    | U+02305      | $\bar{\wedge}$          |
| bbrk;        | U+023B5      | $\cdot\cdot$            |
| bbrktbrk;    | U+023B6      | $\equiv$                |
| bcong;       | U+0224C      | $\cong$                 |
| Bcy;         | U+00411      | Á                       |
| bcy;         | U+00431      | б                       |
| bdquo;       | U+0201E      | „                       |
| becaus;      | U+02235      | $\because$              |
| Because;     | U+02235      | $\because$              |
| because;     | U+02235      | $\because$              |
| bemptyv;     | U+029B0      | $\emptyset$             |
| hepsi;       | U+003F6      | $\epsilon$              |
| bernou;      | U+0212C      | $\mathcal{B}$           |
| Bernoullis;  | U+0212C      | $\mathcal{B}$           |
| Beta;        | U+00392      | B                       |
| beta;        | U+003B2      | $\beta$                 |
| beth;        | U+02136      | $\beth$                 |
| between;     | U+0226C      | $\delta$                |
| Bfr;         | U+1D505      | $\mathfrak{B}$          |

| Nom                 | Personnages)    | Glyphe |
|---------------------|-----------------|--------|
| bfr;                | U+1D51F         | ℓ      |
| bigcap;             | U+022C2         | ∩      |
| bigcirc;            | U+025EF         | ◯      |
| bigcup;             | U+022C3         | ∪      |
| bigodot;            | U+02A00         | ⊙      |
| bigoplus;           | U+02A01         | ⊕      |
| bigotimes;          | U+02A02         | ⊗      |
| bigsqcup;           | U+02A06         | ⊔      |
| bigstar;            | U+02605         | ★      |
| bigtriangledown;    | U+025BD         | ▽      |
| bigtriangleup;      | U+025B3         | △      |
| bigupplus;          | U+02A04         | ⋈      |
| bigvee;             | U+022C1         | ∨      |
| bigwedge;           | U+022C0         | ∧      |
| bkarow;             | U+0290D         | ↞      |
| blacklozenge;       | U+029EB         | ◆      |
| blacksquare;        | U+025AA         | ▪      |
| blacktriangle;      | U+025B4         | ▲      |
| blacktriangledown;  | U+025BE         | ▼      |
| blacktriangleleft;  | U+025C2         | ◀      |
| blacktriangleright; | U+025B8         | ▶      |
| blank;              | U+02423         | ␣      |
| blk12;              | U+02592         | ▤      |
| blk14;              | U+02591         | ▥      |
| blk34;              | U+02593         | ▧      |
| block;              | U+02588         | ■      |
| bne;                | U+0003D U+020E5 | ≠      |
| bnequiv;            | U+02261 U+020E5 | ≢      |
| bNot;               | U+02AED         | ⊐      |
| bnot;               | U+02310         | ¬      |
| Bopf;               | U+1D539         | ℔      |
| bopf;               | U+1D553         | ℔      |
| bot;                | U+022A5         | ⊥      |
| bottom;             | U+022A5         | ⊥      |
| bowtie;             | U+022C8         | ⋈      |

| Nom       | Personnages) | Glyphe |
|-----------|--------------|--------|
| boxbox;   | U+029C9      | 𐄑      |
| boxDL;    | U+02557      | 𐄗      |
| boxDl;    | U+02556      | 𐄖      |
| boxdL;    | U+02555      | 𐄕      |
| boxdl;    | U+02510      | 𐄐      |
| boxDR;    | U+02554      | 𐄔      |
| boxDr;    | U+02553      | 𐄓      |
| boxdR;    | U+02552      | 𐄒      |
| boxdr;    | U+0250C      | 𐄎      |
| boxH;     | U+02550      | =      |
| boxh;     | U+02500      | —      |
| boxHD;    | U+02566      | 𐄞      |
| boxHd;    | U+02564      | 𐄜      |
| boxhD;    | U+02565      | 𐄝      |
| boxhd;    | U+0252C      | 𐄛      |
| boxHU;    | U+02569      | 𐄙      |
| boxHu;    | U+02567      | 𐄟      |
| boxhU;    | U+02568      | 𐄘      |
| boxhu;    | U+02534      | 𐄤      |
| boxminus; | U+0229F      | ⊖      |
| boxplus;  | U+0229E      | ⊕      |
| boxtimes; | U+022A0      | ⊗      |
| boxUL;    | U+0255D      | 𐄛      |
| boxUl;    | U+0255C      | 𐄚      |
| boxuL;    | U+0255B      | 𐄙      |
| boxul;    | U+02518      | 𐄐      |
| boxUR;    | U+0255A      | 𐄔      |
| boxUr;    | U+02559      | 𐄓      |
| boxuR;    | U+02558      | 𐄒      |
| boxur;    | U+02514      | 𐄎      |
| boxV;     | U+02551      |        |
| boxv;     | U+02502      |        |
| boxVH;    | U+0256C      | 𐄟      |
| boxVh;    | U+0256B      | 𐄞      |
| boxvH;    | U+0256A      | 𐄝      |
| boxvh;    | U+0253C      | †      |



| Nom       | Personnages) | Glyphe   |
|-----------|--------------|----------|
| boxVL;    | U+02563      | 𐌚        |
| boxVl;    | U+02562      | 𐌚        |
| boxvL;    | U+02561      | 𐌚        |
| boxvl;    | U+02524      | 𐌚        |
| boxVR;    | U+02560      | 𐌚        |
| boxVr;    | U+0255F      | 𐌚        |
| boxvR;    | U+0255E      | 𐌚        |
| boxvr;    | U+0251C      | 𐌚        |
| bprime;   | U+02035      | ˆ        |
| Breve;    | U+002D8      | ˘        |
| breve;    | U+002D8      | ˘        |
| brvbar;   | U+000A6      | ̂        |
| brvbar    | U+000A6      | ̂        |
| Bscr;     | U+0212C      | <i>ℬ</i> |
| bscr;     | U+1D4B7      | <i>ℬ</i> |
| bsemi;    | U+0204F      | ˙        |
| bsim;     | U+0223D      | ≈        |
| bsime;    | U+022CD      | ≅        |
| bsol;     | U+0005C      | \        |
| bsolb;    | U+029C5      | ⋈        |
| bsolhsub; | U+027C8      | ⋈        |
| bull;     | U+02022      | •        |
| bullet;   | U+02022      | •        |
| bump;     | U+0224E      | ≅        |
| bumpE;    | U+02AAE      | ≅        |
| bumpe;    | U+0224F      | ≅        |
| Bumpeq;   | U+0224E      | ≅        |
| bumpeq;   | U+0224F      | ≅        |
| Cacute;   | U+00106      | Ć        |
| cacute;   | U+00107      | ć        |
| Cap;      | U+022D2      | ⋈        |
| cap;      | U+02229      | ∩        |
| capand;   | U+02A44      | ⋈        |
| capbrcup; | U+02A49      | ⋈        |
| capcap;   | U+02A4B      | ⋈        |
| capcup;   | U+02A47      | ⋈        |

| Nom                   | Personnages)    | Glyphe |
|-----------------------|-----------------|--------|
| capdot;               | U+02A40         | Ṃ      |
| CapitalDifferentialD; | U+02145         | Ɗ      |
| caps;                 | U+02229 U+0FE00 | Რ      |
| caret;                | U+02041         | ˆ      |
| caron;                | U+002C7         | ˇ      |
| Cayleys;              | U+0212D         | ℄      |
| ccaps;                | U+02A4D         | ṃ      |
| Ccaron;               | U+0010C         | Č      |
| ccaron;               | U+0010D         | č      |
| Ccedil;               | U+000C7         | Ç      |
| Ccedil                | U+000C7         | Ç      |
| ccedil;               | U+000E7         | ç      |
| ccedil                | U+000E7         | ç      |
| Ccirc;                | U+00108         | Ĉ      |
| ccirc;                | U+00109         | ĉ      |
| Cconint;              | U+02230         | ∕      |
| ccups;                | U+02A4C         | Ṃ      |
| ccupssm;              | U+02A50         | Ṃ      |
| Cdot;                 | U+0010A         | Ć      |
| cdot;                 | U+0010B         | ć      |
| cedil;                | U+000B8         | ¸      |
| cedil                 | U+000B8         | ¸      |
| Cedilla;              | U+000B8         | ¸      |
| cemptyv;              | U+029B2         | ∅      |
| cent;                 | U+000A2         | ¢      |
| cent                  | U+000A2         | ¢      |
| CenterDot;            | U+000B7         | ·      |
| centerdot;            | U+000B7         | ·      |
| Cfr;                  | U+0212D         | ℄      |
| cfr;                  | U+1D520         | Ꝣ      |
| CHcy;                 | U+00427         | Ң      |
| chcy;                 | U+00447         | ң      |
| check;                | U+02713         | ✓      |
| checkmark;            | U+02713         | ✓      |
| Chi;                  | U+003A7         | Χ      |
| chi;                  | U+003C7         | χ      |

| Nom                       | Personnages) | Glyphe |
|---------------------------|--------------|--------|
| cir;                      | U+025CB      | ○      |
| circ;                     | U+002C6      | ^      |
| circeq;                   | U+02257      | ≐      |
| circlearrowleft;          | U+021BA      | ↶      |
| circlearrowright;         | U+021BB      | ↷      |
| circledast;               | U+0229B      | ⊛      |
| circledcirc;              | U+0229A      | ⊙      |
| circleddash;              | U+0229D      | ⊖      |
| CircleDot;                | U+02299      | ⊙      |
| circledR;                 | U+000AE      | ®      |
| circledS;                 | U+024C8      | Ⓢ      |
| CircleMinus;              | U+02296      | ⊖      |
| CirclePlus;               | U+02295      | ⊕      |
| CircleTimes;              | U+02297      | ⊗      |
| cirE;                     | U+029C3      | ⊖      |
| cire;                     | U+02257      | ≐      |
| cirfnint;                 | U+02A10      | ∫      |
| cirmid;                   | U+02AEF      | ∓      |
| cirscir;                  | U+029C2      | ⊖      |
| ClockwiseContourIntegral; | U+02232      | ∮      |
| CloseCurlyDoubleQuote;    | U+0201D      | ”      |
| CloseCurlyQuote;          | U+02019      | '      |
| clubs;                    | U+02663      | ♣      |
| clubsuit;                 | U+02663      | ♣      |
| Colon;                    | U+02237      | ::     |
| colon;                    | U+0003A      | :      |
| Colone;                   | U+02A74      | ::=    |
| colone;                   | U+02254      | :=     |
| coloneq;                  | U+02254      | :=     |
| comma;                    | U+0002C      | ,      |
| commat;                   | U+00040      | @      |
| comp;                     | U+02201      | ℂ      |
| compfn;                   | U+02218      | ◦      |
| complement;               | U+02201      | ℂ      |
| complexes;                | U+02102      | ℂ      |
| cong;                     | U+02245      | ≅      |

| Nom                              | Personnages) | Glyphe             |
|----------------------------------|--------------|--------------------|
| congdot;                         | U+02A6D      | $\cong$            |
| Congruent;                       | U+02261      | $\equiv$           |
| Conint;                          | U+0222F      | $\oint$            |
| conint;                          | U+0222E      | $\oint$            |
| ContourIntegral;                 | U+0222E      | $\oint$            |
| Copf;                            | U+02102      | $\mathbb{C}$       |
| copf;                            | U+1D554      | $\mathfrak{C}$     |
| coprod;                          | U+02210      | $\amalg$           |
| Coproduct;                       | U+02210      | $\amalg$           |
| COPY;                            | U+000A9      | ©                  |
| COPY                             | U+000A9      | ©                  |
| copy;                            | U+000A9      | ©                  |
| copy                             | U+000A9      | ©                  |
| copysr;                          | U+02117      | ®                  |
| CounterClockwiseContourIntegral; | U+02233      | $\oint$            |
| crarr;                           | U+021B5      | $\leftarrow$       |
| Cross;                           | U+02A2F      | $\times$           |
| cross;                           | U+02717      | $\times$           |
| Cscr;                            | U+1D49E      | $\mathcal{C}$      |
| cscr;                            | U+1D4B8      | $\mathfrak{c}$     |
| csub;                            | U+02ACF      | $\sqsubset$        |
| csube;                           | U+02AD1      | $\sqsubseteq$      |
| csup;                            | U+02AD0      | $\sqsupset$        |
| csupe;                           | U+02AD2      | $\sqsupseteq$      |
| ctdot;                           | U+022EF      | $\cdots$           |
| cudarrrl;                        | U+02938      | $\curvearrowright$ |
| cudarrrr;                        | U+02935      | $\curvearrowleft$  |
| cuepr;                           | U+022DE      | $\preccurlyeq$     |
| cuesc;                           | U+022DF      | $\succcurlyeq$     |
| cularr;                          | U+021B6      | $\curvearrowright$ |
| cularrp;                         | U+0293D      | $\curvearrowleft$  |
| Cup;                             | U+022D3      | $\sqcup$           |
| cup;                             | U+0222A      | $\cup$             |
| cupbrcap;                        | U+02A48      | $\bowtie$          |
| CupCap;                          | U+0224D      | $\bowtie$          |
| cupcap;                          | U+02A46      | $\bowtie$          |

| Nom              | Personnages)    | Glyphe |
|------------------|-----------------|--------|
| cupcup;          | U+02A4A         | Ƶ      |
| cupdot;          | U+0228D         | Ʊ      |
| cupor;           | U+02A45         | Ʈ      |
| cups;            | U+0222A U+0FE00 | Ʋ      |
| curarr;          | U+021B7         | ↷      |
| curarrm;         | U+0293C         | ↷      |
| curlyeqprec;     | U+022DE         | ⋞      |
| curlyeqsucc;     | U+022DF         | ⋟      |
| curlyvee;        | U+022CE         | ⋎      |
| curlywedge;      | U+022CF         | ⋏      |
| curren;          | U+000A4         | ¤      |
| curren           | U+000A4         | ¤      |
| curvearrowleft;  | U+021B6         | ↶      |
| curvearrowright; | U+021B7         | ↷      |
| cuvee;           | U+022CE         | ⋎      |
| cuwed;           | U+022CF         | ⋏      |
| cwconint;        | U+02232         | ∯      |
| cwint;           | U+02231         | ∮      |
| cylcty;          | U+0232D         | ◻      |
| Dagger;          | U+02021         | ‡      |
| dagger;          | U+02020         | †      |
| daleth;          | U+02138         | ד      |
| Darr;            | U+021A1         | ↴      |
| dArr;            | U+021D3         | ↵      |
| darr;            | U+02193         | ↴      |
| dash;            | U+02010         | -      |
| Dashv;           | U+02AE4         | ≡      |
| dashv;           | U+022A3         | ⊣      |
| dbkarow;         | U+0290F         | ↔      |
| dblac;           | U+002DD         | ¨      |
| Dcaron;          | U+0010E         | Ď      |
| dcaron;          | U+0010F         | ď      |
| Dcy;             | U+00414         | Ä      |
| dcy;             | U+00434         | ä      |
| DD;              | U+02145         | Ɔ      |
| dd;              | U+02146         | ɔ      |

| Nom                     | Personnages) | Glyphe   |
|-------------------------|--------------|----------|
| ddagger;                | U+02021      | ‡        |
| ddarr;                  | U+021CA      | ⇓        |
| DDottrahd;              | U+02911      | ⋯→       |
| ddotseq;                | U+02A77      | ≐        |
| deg;                    | U+000B0      | °        |
| deg                     | U+000B0      | °        |
| Del;                    | U+02207      | ∇        |
| Delta;                  | U+00394      | Δ        |
| delta;                  | U+003B4      | δ        |
| demptyv;                | U+029B1      | ∅        |
| dfisht;                 | U+0297F      | Ⅎ        |
| Dfr;                    | U+1D507      | 𝔻        |
| dfr;                    | U+1D521      | 𝔡        |
| dHar;                   | U+02965      | ⇓        |
| dharl;                  | U+021C3      | ⋇        |
| dharr;                  | U+021C2      | ⋆        |
| DiacriticalAcute;       | U+000B4      | ´        |
| DiacriticalDot;         | U+002D9      | ·        |
| DiacriticalDoubleAcute; | U+002DD      | ¨        |
| DiacriticalGrave;       | U+00060      | `        |
| DiacriticalTilde;       | U+002DC      | ˜        |
| diam;                   | U+022C4      | ◊        |
| Diamond;                | U+022C4      | ◊        |
| diamond;                | U+022C4      | ◊        |
| diamondsuit;            | U+02666      | ◆        |
| diams;                  | U+02666      | ◆        |
| die;                    | U+000A8      | ¨        |
| DifferentialD;          | U+02146      | <i>d</i> |
| digamma;                | U+003DD      | Ϝ        |
| disin;                  | U+022F2      | €        |
| div;                    | U+000F7      | ÷        |
| divide;                 | U+000F7      | ÷        |
| divide                  | U+000F7      | ÷        |
| divideontimes;          | U+022C7      | ⋈        |
| divonx;                 | U+022C7      | ⋈        |
| DJcy;                   | U+00402      | Ђ        |

| Nom                       | Personnages) | Glyphe |
|---------------------------|--------------|--------|
| djcy;                     | U+00452      | ḣ      |
| dlcorn;                   | U+0231E      | ⌞      |
| dlcrop;                   | U+0230D      | ⌟      |
| dollar;                   | U+00024      | \$     |
| Dopf;                     | U+1D53B      | ℰ      |
| dopf;                     | U+1D555      | ℄      |
| Dot;                      | U+000A8      | ¨      |
| dot;                      | U+002D9      | ·      |
| DotDot;                   | U+020DC      | ⋯      |
| doteq;                    | U+02250      | ≐      |
| doteqdot;                 | U+02251      | ≑      |
| DotEqual;                 | U+02250      | ≐      |
| dotminus;                 | U+02238      | ⋈      |
| dotplus;                  | U+02214      | ⋉      |
| dotsquare;                | U+022A1      | ⊠      |
| doublebarwedge;           | U+02306      | ⋈      |
| DoubleContourIntegral;    | U+0222F      | ⧻      |
| DoubleDot;                | U+000A8      | ¨      |
| DoubleDownArrow;          | U+021D3      | ⇓      |
| DoubleLeftArrow;          | U+021D0      | ⇐      |
| DoubleLeftRightArrow;     | U+021D4      | ⇔      |
| DoubleLeftTee;            | U+02AE4      | ⇐      |
| DoubleLongLeftArrow;      | U+027F8      | ⇐      |
| DoubleLongLeftRightArrow; | U+027FA      | ⇔      |
| DoubleLongRightArrow;     | U+027F9      | ⇒      |
| DoubleRightArrow;         | U+021D2      | ⇒      |
| DoubleRightTee;           | U+022A8      | ⇐      |
| DoubleUpArrow;            | U+021D1      | ⇑      |
| DoubleUpDownArrow;        | U+021D5      | ⇕      |
| DoubleVerticalBar;        | U+02225      |        |
| DownArrow;                | U+02193      | ↓      |
| Downarrow;                | U+021D3      | ⇓      |
| downarrow;                | U+02193      | ↓      |
| DownArrowBar;             | U+02913      | ⬇      |
| DownArrowUpArrow;         | U+021F5      | ⇕      |
| DownBreve;                | U+00311      | ḡ      |

| Nom                  | Personnages) | Glyphe   |
|----------------------|--------------|----------|
| downdownarrows;      | U+021CA      | ⇓        |
| downharpoonleft;     | U+021C3      | ↵        |
| downharpoonright;    | U+021C2      | ↘        |
| DownLeftRightVector; | U+02950      | ↗        |
| DownLeftTeeVector;   | U+0295E      | ↖        |
| DownLeftVector;      | U+021BD      | ↖        |
| DownLeftVectorBar;   | U+02956      | ↖        |
| DownRightTeeVector;  | U+0295F      | ↗        |
| DownRightVector;     | U+021C1      | ↗        |
| DownRightVectorBar;  | U+02957      | ↗        |
| DownTee;             | U+022A4      | ⌞        |
| DownTeeArrow;        | U+021A7      | ⇓        |
| drbkarow;            | U+02910      | ↗↘       |
| drcorn;              | U+0231F      | ┐        |
| drcrop;              | U+0230C      | ┐        |
| Dscr;                | U+1D49F      | <i>ℒ</i> |
| dscr;                | U+1D4B9      | <i>ℓ</i> |
| DScy;                | U+00405      | Ѕ        |
| dscy;                | U+00455      | ѕ        |
| dsol;                | U+029F6      | ↗        |
| Dstrok;              | U+00110      | Đ        |
| dstrok;              | U+00111      | đ        |
| dtdot;               | U+022F1      | ⋮        |
| dtri;                | U+025BF      | ▼        |
| dtrif;               | U+025BE      | ▼        |
| duarr;               | U+021F5      | ⇕        |
| duhar;               | U+0296F      | ⇕        |
| dwangle;             | U+029A6      | ↘        |
| DZcy;                | U+0040F      | џ        |
| dzcy;                | U+0045F      | ѣ        |
| dzigrrarr;           | U+027FF      | ↗↘       |
| Eacute;              | U+000C9      | É        |
| Eacute               | U+000C9      | É        |
| eacute;              | U+000E9      | é        |
| eacute               | U+000E9      | é        |
| easter;              | U+02A6E      | ⌘        |



| Nom               | Personnages) | Glyphe |
|-------------------|--------------|--------|
| Ecaron;           | U+0011A      | Ě      |
| ecaron;           | U+0011B      | ě      |
| ecir;             | U+02256      | ⌘      |
| Ecirc;            | U+000CA      | Ê      |
| Ecirc             | U+000CA      | Ê      |
| ecirc;            | U+000EA      | ê      |
| ecirc             | U+000EA      | ê      |
| ecolon;           | U+02255      | ⌚      |
| Ecy;              | U+0042D      | Ý      |
| ecy;              | U+0044D      | í      |
| eDDot;            | U+02A77      | ⌛      |
| Edot;             | U+00116      | Ė      |
| eDot;             | U+02251      | ⌙      |
| edot;             | U+00117      | ì      |
| ee;               | U+02147      | ℓ      |
| efDot;            | U+02252      | ⌚      |
| Efr;              | U+1D508      | ℔      |
| efr;              | U+1D522      | e      |
| eg;               | U+02A9A      | ⌞      |
| Egrave;           | U+000C8      | È      |
| Egrave            | U+000C8      | È      |
| egrave;           | U+000E8      | è      |
| egrave            | U+000E8      | è      |
| egs;              | U+02A96      | ⌞      |
| egsdot;           | U+02A98      | ⌞      |
| el;               | U+02A99      | ⌞      |
| Element;          | U+02208      | €      |
| elinters;         | U+023E7      | □      |
| ell;              | U+02113      | ℓ      |
| els;              | U+02A95      | ⌞      |
| elsdot;           | U+02A97      | ⌞      |
| Emacr;            | U+00112      | Ě      |
| emacr;            | U+00113      | ě      |
| empty;            | U+02205      | ∅      |
| emptyset;         | U+02205      | ∅      |
| EmptySmallSquare; | U+025FB      | □      |

| Nom                   | Personnages) | Glyphe |
|-----------------------|--------------|--------|
| emptyv;               | U+02205      | ∅      |
| EmptyVerySmallSquare; | U+025AB      | ◻      |
| emsp;                 | U+02003      |        |
| emsp13;               | U+02004      |        |
| emsp14;               | U+02005      |        |
| ENG;                  | U+0014A      | Ŋ      |
| eng;                  | U+0014B      | ŋ      |
| ensp;                 | U+02002      |        |
| Eogon;                | U+00118      | Ê      |
| eogon;                | U+00119      | Ė      |
| Eopf;                 | U+1D53C      | ℰ      |
| eopf;                 | U+1D556      | ℰ      |
| epar;                 | U+022D5      | #      |
| eparsl;               | U+029E3      | #      |
| eplus;                | U+02A71      | ≡      |
| epsi;                 | U+003B5      | ε      |
| Epsilon;              | U+00395      | E      |
| epsilon;              | U+003B5      | ε      |
| epsiv;                | U+003F5      | ϵ      |
| eqcirc;               | U+02256      | ⊖      |
| eqcolon;              | U+02255      | ⋮      |
| eqsim;                | U+02242      | ≈      |
| eqslantgtr;           | U+02A96      | ⋙      |
| eqslantless;          | U+02A95      | ⋘      |
| Equal;                | U+02A75      | ⋈      |
| equals;               | U+0003D      | =      |
| EqualTilde;           | U+02242      | ≈      |
| equest;               | U+0225F      | ⋮      |
| Equilibrium;          | U+021CC      | ⇌      |
| equiv;                | U+02261      | ≡      |
| equivDD;              | U+02A78      | ≡      |
| eqvparsl;             | U+029E5      | #      |
| erarr;                | U+02971      | ⇒      |
| erDot;                | U+02253      | ≡      |
| Escr;                 | U+02130      | ℰ      |
| escr;                 | U+0212F      | e      |

| Nom                    | Personnages)    | Glyphe   |
|------------------------|-----------------|----------|
| esdot;                 | U+02250         | ≐        |
| Esim;                  | U+02A73         | ≍        |
| esim;                  | U+02242         | ≈        |
| Eta;                   | U+00397         | Η        |
| eta;                   | U+003B7         | η        |
| ETH;                   | U+000D0         | Đ        |
| ETH                    | U+000D0         | Đ        |
| eth;                   | U+000F0         | đ        |
| eth                    | U+000F0         | đ        |
| Euml;                  | U+000CB         | Ë        |
| Euml                   | U+000CB         | Ë        |
| euml;                  | U+000EB         | ë        |
| euml                   | U+000EB         | ë        |
| euro;                  | U+020AC         | €        |
| excl;                  | U+00021         | !        |
| exist;                 | U+02203         | ∃        |
| Exists;                | U+02203         | ∃        |
| expectation;           | U+02130         | ℰ        |
| ExponentialE;          | U+02147         | <i>e</i> |
| exponentiale;          | U+02147         | <i>e</i> |
| fallingdotseq;         | U+02252         | ≒        |
| Fcy;                   | U+00424         | Φ        |
| fcy;                   | U+00444         | φ        |
| female;                | U+02640         | ♀        |
| ffilig;                | U+0FB03         | ffi      |
| fflig;                 | U+0FB00         | ff       |
| ffllig;                | U+0FB04         | ffl      |
| Ffr;                   | U+1D509         | ℱ        |
| ffr;                   | U+1D523         | ƒ        |
| filig;                 | U+0FB01         | fi       |
| FilledSmallSquare;     | U+025FC         | ■        |
| FilledVerySmallSquare; | U+025AA         | ▪        |
| fjlig;                 | U+00066 U+0006A | fj       |
| flat;                  | U+0266D         | ♭        |
| fllig;                 | U+0FB02         | fl       |
| fltns;                 | U+025B1         | ▭        |

| Nom         | Personnages) | Glyphe         |
|-------------|--------------|----------------|
| fnof;       | U+00192      | <i>f</i>       |
| Fopf;       | U+1D53D      | ℱ              |
| fopf;       | U+1D557      | ff             |
| ForAll;     | U+02200      | ∀              |
| forall;     | U+02200      | ∀              |
| fork;       | U+022D4      | 𐍋              |
| forkv;      | U+02AD9      | 𐍉              |
| Fouriertrf; | U+02131      | $\mathcal{F}$  |
| fpartint;   | U+02A0D      | $\int$         |
| frac12;     | U+000BD      | $\frac{1}{2}$  |
| frac12      | U+000BD      | $\frac{1}{2}$  |
| frac13;     | U+02153      | $\frac{1}{3}$  |
| frac14;     | U+000BC      | $\frac{1}{4}$  |
| frac14      | U+000BC      | $\frac{1}{4}$  |
| frac15;     | U+02155      | $\frac{1}{5}$  |
| frac16;     | U+02159      | $\frac{1}{6}$  |
| frac18;     | U+0215B      | $\frac{1}{8}$  |
| frac23;     | U+02154      | $\frac{2}{3}$  |
| frac25;     | U+02156      | $\frac{2}{5}$  |
| frac34;     | U+000BE      | $\frac{3}{4}$  |
| frac34      | U+000BE      | $\frac{3}{4}$  |
| frac35;     | U+02157      | $\frac{3}{5}$  |
| frac38;     | U+0215C      | $\frac{3}{8}$  |
| frac45;     | U+02158      | $\frac{4}{5}$  |
| frac56;     | U+0215A      | $\frac{5}{6}$  |
| frac58;     | U+0215D      | $\frac{5}{8}$  |
| frac78;     | U+0215E      | $\frac{7}{8}$  |
| frasl;      | U+02044      | /              |
| frown;      | U+02322      | ⤵              |
| Fscr;       | U+02131      | $\mathcal{F}$  |
| fscr;       | U+1D4BB      | $\mathfrak{f}$ |
| gacute;     | U+001F5      | ǧ              |
| Gamma;      | U+00393      | Γ              |
| gamma;      | U+003B3      | γ              |
| Gammad;     | U+003DC      | Ɔ              |
| gammad;     | U+003DD      | Ɔ              |

| Nom       | Personnages)    | Glyphe |
|-----------|-----------------|--------|
| gap;      | U+02A86         | ≈      |
| Gbreve;   | U+0011E         | G      |
| gbreve;   | U+0011F         | g      |
| Gcedil;   | U+00122         | G      |
| Gcirc;    | U+0011C         | G      |
| gcirc;    | U+0011D         | g      |
| Gcy;      | U+00413         | Ă      |
| gcy;      | U+00433         | Ț      |
| Gdot;     | U+00120         | G      |
| gdot;     | U+00121         | g      |
| gE;       | U+02267         | ≧      |
| ge;       | U+02265         | ≥      |
| gEl;      | U+02A8C         | ≧̸     |
| gel;      | U+022DB         | ≧̸     |
| geq;      | U+02265         | ≥      |
| geqq;     | U+02267         | ≧      |
| geqslant; | U+02A7E         | ≱     |
| ges;      | U+02A7E         | ≱     |
| gescc;    | U+02AA9         | ▷̸     |
| gesdot;   | U+02A80         | ≱     |
| gesdoto;  | U+02A82         | ≱     |
| gesdotol; | U+02A84         | ≱     |
| gesl;     | U+022DB U+0FE00 | ≧̸     |
| gesles;   | U+02A94         | ≧̸     |
| Gfr;      | U+1D50A         | ℱ      |
| gfr;      | U+1D524         | g      |
| Gg;       | U+022D9         | ≫      |
| gg;       | U+0226B         | ≫      |
| ggg;      | U+022D9         | ≫      |
| gimel;    | U+02137         | ℘      |
| GJcy;     | U+00403         | Ĳ      |
| gjcy;     | U+00453         | ŕ      |
| gl;       | U+02277         | ≧̸     |
| gla;      | U+02AA5         | ><     |
| glE;      | U+02A92         | ≧̸     |
| glj;      | U+02AA4         | ⌘      |

| Nom                | Personnages) | Glyphe |
|--------------------|--------------|--------|
| gnap;              | U+02A8A      | ≈      |
| gnapprox;          | U+02A8A      | ≈      |
| gnE;               | U+02269      | ≇      |
| gne;               | U+02A88      | ≧      |
| gneq;              | U+02A88      | ≧      |
| gneqq;             | U+02269      | ≇      |
| gnsim;             | U+022E7      | ≈      |
| Gopf;              | U+1D53E      | ℔      |
| gopf;              | U+1D558      | ℔      |
| grave;             | U+00060      | `      |
| GreaterEqual;      | U+02265      | ≥      |
| GreaterEqualLess;  | U+022DB      | ≧      |
| GreaterFullEqual;  | U+02267      | ≡      |
| GreaterGreater;    | U+02AA2      | ≫      |
| GreaterLess;       | U+02277      | ≧      |
| GreaterSlantEqual; | U+02A7E      | ≧      |
| GreaterTilde;      | U+02273      | ≈      |
| Gscr;              | U+1D4A2      | ℊ      |
| gscr;              | U+0210A      | g      |
| gsim;              | U+02273      | ≈      |
| gsime;             | U+02A8E      | ≧      |
| gsiml;             | U+02A90      | ≈      |
| GT;                | U+0003E      | >      |
| GT                 | U+0003E      | >      |
| Gt;                | U+0226B      | ≫      |
| gt;                | U+0003E      | >      |
| gt                 | U+0003E      | >      |
| gtcc;              | U+02AA7      | ▷      |
| gtcir;             | U+02A7A      | ▷      |
| gtdot;             | U+022D7      | ▷      |
| gtlPar;            | U+02995      | ⋈      |
| gtquest;           | U+02A7C      | ▷      |
| gtrapprox;         | U+02A86      | ≈      |
| gtrarr;            | U+02978      | ⇒      |
| gtrdot;            | U+022D7      | ▷      |
| gtreqless;         | U+022DB      | ≧      |

| Nom             | Personnages)    | Glyphe            |
|-----------------|-----------------|-------------------|
| gtreqqless;     | U+02A8C         | $\gtrless$        |
| gtrless;        | U+02277         | $\gtr$            |
| gtrsim;         | U+02273         | $\gtrsim$         |
| gvertneqq;      | U+02269 U+0FE00 | $\gtrless$        |
| gvnE;           | U+02269 U+0FE00 | $\gtrless$        |
| Hacek;          | U+002C7         | ˇ                 |
| hairsp;         | U+0200A         |                   |
| half;           | U+000BD         | $\frac{1}{2}$     |
| hamilt;         | U+0210B         | $\mathcal{H}$     |
| HARDcy;         | U+0042A         | Ђ                 |
| hardcy;         | U+0044A         | ѓ                 |
| hArr;           | U+021D4         | $\Leftrightarrow$ |
| harr;           | U+02194         | $\leftrightarrow$ |
| harrcir;        | U+02948         | $\Leftrightarrow$ |
| harrw;          | U+021AD         | $\Leftrightarrow$ |
| Hat;            | U+0005E         | ^                 |
| hbar;           | U+0210F         | $\hbar$           |
| Hcirc;          | U+00124         | $\hat{H}$         |
| hcirc;          | U+00125         | $\hat{h}$         |
| hearts;         | U+02665         | ♥                 |
| heartsuit;      | U+02665         | ♥                 |
| hellip;         | U+02026         | ...               |
| hercon;         | U+022B9         | ÷                 |
| Hfr;            | U+0210C         | ℋ                 |
| hfr;            | U+1D525         | ℏ                 |
| HilbertSpace;   | U+0210B         | $\mathcal{H}$     |
| hksearow;       | U+02925         | ↷                 |
| hkswarow;       | U+02926         | ↶                 |
| hoarr;          | U+021FF         | $\leftrightarrow$ |
| homtht;         | U+0223B         | $\approx$         |
| hookleftarrow;  | U+021A9         | $\hookleftarrow$  |
| hookrightarrow; | U+021AA         | $\hookrightarrow$ |
| Hopf;           | U+0210D         | H                 |
| hopf;           | U+1D559         | ℏ                 |
| horbar;         | U+02015         | —                 |
| HorizontalLine; | U+02500         | —                 |

| Nom           | Personnages) | Glyphe            |
|---------------|--------------|-------------------|
| Hscr;         | U+0210B      | $\mathcal{H}$     |
| hscr;         | U+1D4BD      | $\mathfrak{h}$    |
| hslash;       | U+0210F      | $\hbar$           |
| Hstrook;      | U+00126      | H                 |
| hstrook;      | U+00127      | h                 |
| HumpDownHump; | U+0224E      | $\mathfrak{Z}$    |
| HumpEqual;    | U+0224F      | $\mathfrak{Z}$    |
| hybull;       | U+02043      | -                 |
| hyphen;       | U+02010      | -                 |
| Iacute;       | U+000CD      | JE                |
| Iacute        | U+000CD      | JE                |
| iacute;       | U+000ED      | je                |
| iacute        | U+000ED      | je                |
| ic;           | U+02063      | $\mathfrak{I}$    |
| Icirc;        | U+000CE      | JE                |
| Icirc         | U+000CE      | JE                |
| icirc;        | U+000EE      | je                |
| icirc         | U+000EE      | je                |
| Icy;          | U+00418      | È                 |
| icy;          | U+00438      | è                 |
| Idot;         | U+00130      | JE                |
| IEcy;         | U+00415      | Å                 |
| iecy;         | U+00435      | e                 |
| isexcl;       | U+000A1      | i                 |
| isexcl        | U+000A1      | i                 |
| iff;          | U+021D4      | $\Leftrightarrow$ |
| Ifr;          | U+02111      | $\mathfrak{I}$    |
| ifr;          | U+1D526      | i                 |
| Igrave;       | U+000CC      | JE                |
| Igrave        | U+000CC      | JE                |
| igrave;       | U+000EC      | je                |
| igrave        | U+000EC      | je                |
| ii;           | U+02148      | $\mathfrak{i}$    |
| iiiint;       | U+02A0C      | $\iiint$          |
| iiint;        | U+0222D      | $\mathfrak{f}$    |
| iinfin;       | U+029DC      | $\omega$          |



| Nom             | Personnages) | Glyphe   |
|-----------------|--------------|----------|
| iota;           | U+02129      | ı        |
| IJlig;          | U+00132      | IJ       |
| ijlig;          | U+00133      | ij       |
| Im;             | U+02111      | ℑ        |
| Imacr;          | U+0012A      | JE       |
| imacr;          | U+0012B      | je       |
| image;          | U+02111      | ℑ        |
| ImaginaryI;     | U+02148      | <i>i</i> |
| imagline;       | U+02110      | <i>ℑ</i> |
| imagpart;       | U+02111      | ℑ        |
| imath;          | U+00131      | je       |
| imof;           | U+022B7      | •∞       |
| imped;          | U+001B5      | Z        |
| Implies;        | U+021D2      | ⇒        |
| in;             | U+02208      | ∈        |
| incare;         | U+02105      | %        |
| infin;          | U+0221E      | ∞        |
| infintie;       | U+029DD      | ∞        |
| inodot;         | U+00131      | je       |
| Int;            | U+0222C      | ∬        |
| int;            | U+0222B      | ∫        |
| intcal;         | U+022BA      | ⊢        |
| integers;       | U+02124      | Z        |
| Integral;       | U+0222B      | ∫        |
| intercal;       | U+022BA      | ⊢        |
| Intersection;   | U+022C2      | ∩        |
| intlarkhk;      | U+02A17      | ℥        |
| intprod;        | U+02A3C      | ⊣        |
| InvisibleComma; | U+02063      | ␣        |
| InvisibleTimes; | U+02062      | ␣        |
| IOcy;           | U+00401      | Ë        |
| iocy;           | U+00451      | ë        |
| Iogon;          | U+0012E      | JE       |
| iogon;          | U+0012F      | je       |
| Iopf;           | U+1D540      | Ⅎ        |
| iopf;           | U+1D55A      | Ⅎ        |

| Nom      | Personnages) | Glyphe   |
|----------|--------------|----------|
| Iota;    | U+00399      | je       |
| iota;    | U+003B9      | ı        |
| iprod;   | U+02A3C      | ┘        |
| iquest;  | U+000BF      | ı        |
| iquest   | U+000BF      | ı        |
| Iscr;    | U+02110      | <i>ℒ</i> |
| iscr;    | U+1D4BE      | <i>i</i> |
| isin;    | U+02208      | €        |
| isindot; | U+022F5      | €̇       |
| isinE;   | U+022F9      | €̂       |
| isins;   | U+022F4      | €̣       |
| isinsv;  | U+022F3      | €̤       |
| isinv;   | U+02208      | €        |
| it;      | U+02062      | ⌘        |
| Itilde;  | U+00128      | JE       |
| itilde;  | U+00129      | je       |
| Iukcy;   | U+00406      | I        |
| iukcy;   | U+00456      | dans     |
| Iuml;    | U+000CF      | JE       |
| Iuml     | U+000CF      | JE       |
| iuml;    | U+000EF      | je       |
| iuml     | U+000EF      | je       |
| Jcirc;   | U+00134      | ĵ        |
| jcirc;   | U+00135      | ĵ        |
| Jcy;     | U+00419      | Љ        |
| jcy;     | U+00439      | ©        |
| Jfr;     | U+1D50D      | ℑ        |
| jfr;     | U+1D527      | ı        |
| jmath;   | U+00237      | j        |
| Jopf;    | U+1D541      | ℐ        |
| jopf;    | U+1D55B      | ℐ        |
| Jscr;    | U+1D4A5      | <i>ℒ</i> |
| jscr;    | U+1D4BF      | <i>ℒ</i> |
| Jsercy;  | U+00408      | J        |
| jsercy;  | U+00458      | j        |
| Jukcy;   | U+00404      | €        |

| Nom          | Personnages) | Glyphe |
|--------------|--------------|--------|
| jukcy;       | U+00454      | ϵ      |
| Kappa;       | U+0039A      | Κ      |
| kappa;       | U+003BA      | κ      |
| kappav;      | U+003F0      | Ϡ      |
| Kcedil;      | U+00136      | Ḳ      |
| kcedil;      | U+00137      | ḳ      |
| Kcy;         | U+0041A      | Ɔ      |
| kcy;         | U+0043A      | ƿ      |
| Kfr;         | U+1D50E      | Ɔ      |
| kfr;         | U+1D528      | ƿ      |
| kgreen;      | U+00138      | κ      |
| KHcy;        | U+00425      | X      |
| khcy;        | U+00445      | x      |
| KJcy;        | U+0040C      | Ć      |
| kjcy;        | U+0045C      | ć      |
| Kopf;        | U+1D542      | ℔      |
| kopf;        | U+1D55C      | ℔      |
| Kscr;        | U+1D4A6      | ℔      |
| kscr;        | U+1D4C0      | ℔      |
| lAarr;       | U+021DA      | ⇐      |
| Lacute;      | U+00139      | Ĺ      |
| lacute;      | U+0013A      | ĺ      |
| laemptyv;    | U+029B4      | ∅      |
| lagran;      | U+02112      | ℒ      |
| Lambda;      | U+0039B      | Λ      |
| lambda;      | U+003BB      | λ      |
| Lang;        | U+027EA      | ⟨      |
| lang;        | U+027E8      | ⟨      |
| langd;       | U+02991      | ⟨      |
| langle;      | U+027E8      | ⟨      |
| lap;         | U+02A85      | ≈      |
| Laplacetrfr; | U+02112      | ℒ      |
| laquo;       | U+000AB      | «      |
| laquo        | U+000AB      | «      |
| Larr;        | U+0219E      | ⇐      |
| lArr;        | U+021D0      | ⇐      |

| Nom       | Personnages)    | Glyphe |
|-----------|-----------------|--------|
| larr;     | U+02190         | ←      |
| larrb;    | U+021E4         | ↵      |
| larrbfs;  | U+0291F         | ↵      |
| larrfs;   | U+0291D         | ↵      |
| larrhk;   | U+021A9         | ↶      |
| larrlp;   | U+021AB         | ↶      |
| larrpl;   | U+02939         | ↷      |
| larrsim;  | U+02973         | ↷      |
| larrtl;   | U+021A2         | ↷      |
| lat;      | U+02AAB         | ➤      |
| lAtail;   | U+0291B         | ↵      |
| latail;   | U+02919         | ↵      |
| late;     | U+02AAD         | ➦      |
| lates;    | U+02AAD U+0FE00 | ➦      |
| lBarr;    | U+0290E         | ↵      |
| lbarr;    | U+0290C         | ↵      |
| lbbrk;    | U+02772         | (      |
| lbrace;   | U+0007B         | {      |
| lbrack;   | U+0005B         | [      |
| lbrke;    | U+0298B         | [      |
| lbrksld;  | U+0298F         | [      |
| lbrkslu;  | U+0298D         | [      |
| Lcaron;   | U+0013D         | Ľ      |
| lcaron;   | U+0013E         | ¾      |
| Lcedil;   | U+0013B         | Ł      |
| lcedil;   | U+0013C         | ĩ      |
| lceil;    | U+02308         | ⌈      |
| lcub;     | U+0007B         | {      |
| Lcy;      | U+0041B         | Ё      |
| lcy;      | U+0043B         | Ѡ      |
| ldca;     | U+02936         | ↶      |
| ldquo;    | U+0201C         | "      |
| ldquor;   | U+0201E         | „      |
| ldrdhar;  | U+02967         | ↷      |
| ldrushar; | U+0294B         | ↶      |
| ldsh;     | U+021B2         | ↶      |

| Nom                  | Personnages) | Glyphe |
|----------------------|--------------|--------|
| lE;                  | U+02266      | ≤      |
| le;                  | U+02264      | ≤      |
| LeftAngleBracket;    | U+027E8      | ⟨      |
| LeftArrow;           | U+02190      | ←      |
| Leftarrow;           | U+021D0      | ⇐      |
| leftarrow;           | U+02190      | ←      |
| LeftArrowBar;        | U+021E4      | ↵      |
| LeftArrowRightArrow; | U+021C6      | ↔      |
| leftarrowtail;       | U+021A2      | ↵      |
| LeftCeiling;         | U+02308      | ⌈      |
| LeftDoubleBracket;   | U+027E6      | ⟦      |
| LeftDownTeeVector;   | U+02961      | ⌋      |
| LeftDownVector;      | U+021C3      | ⌋      |
| LeftDownVectorBar;   | U+02959      | ⌋      |
| LeftFloor;           | U+0230A      | ⌋      |
| leftharpoondown;     | U+021BD      | ↵      |
| leftharpoonup;       | U+021BC      | ↵      |
| leftleftarrows;      | U+021C7      | ⇐      |
| LeftRightArrow;      | U+02194      | ↔      |
| Leftrightarrow;      | U+021D4      | ⇔      |
| leftrightharpoonup;  | U+02194      | ↔      |
| leftrightharpoons;   | U+021C6      | ↔      |
| leftrightharpoons;   | U+021CB      | ↔      |
| leftrightsquigarrow; | U+021AD      | ↗      |
| LeftRightVector;     | U+0294E      | ↔      |
| LeftTee;             | U+022A3      | ⊣      |
| LeftTeeArrow;        | U+021A4      | ↵      |
| LeftTeeVector;       | U+0295A      | ↵      |
| leftthreetimes;      | U+022CB      | ⋈      |
| LeftTriangle;        | U+022B2      | ◁      |
| LeftTriangleBar;     | U+029CF      | ◁      |
| LeftTriangleEqual;   | U+022B4      | ◁      |
| LeftUpDownVector;    | U+02951      | ↕      |
| LeftUpTeeVector;     | U+02960      | ⌋      |
| LeftUpVector;        | U+021BF      | ⌋      |
| LeftUpVectorBar;     | U+02958      | ⌋      |

| Nom               | Personnages)    | Glyphe |
|-------------------|-----------------|--------|
| LeftVector;       | U+021BC         | ←      |
| LeftVectorBar;    | U+02952         | ↵      |
| lEg;              | U+02A8B         | ↯      |
| leg;              | U+022DA         | ↯      |
| leq;              | U+02264         | ≤      |
| leqq;             | U+02266         | ≐      |
| leqslant;         | U+02A7D         | ⩵      |
| les;              | U+02A7D         | ⩵      |
| lescc;            | U+02AA8         | ⩶      |
| lesdot;           | U+02A7F         | ⩷      |
| lesdoto;          | U+02A81         | ⩸      |
| lesdotor;         | U+02A83         | ⩹      |
| lesg;             | U+022DA U+0FE00 | ↯      |
| lesges;           | U+02A93         | ↯      |
| lessapprox;       | U+02A85         | ⩺      |
| lessdot;          | U+022D6         | ⩴      |
| lesseqgtr;        | U+022DA         | ↯      |
| lesseqqgtr;       | U+02A8B         | ↯      |
| LessEqualGreater; | U+022DA         | ↯      |
| LessFullEqual;    | U+02266         | ≐      |
| LessGreater;      | U+02276         | ⩵      |
| lessgtr;          | U+02276         | ⩵      |
| LessLess;         | U+02AA1         | ⩴      |
| lesssim;          | U+02272         | ⩻      |
| LessSlantEqual;   | U+02A7D         | ⩵      |
| LessTilde;        | U+02272         | ⩻      |
| lfisht;           | U+0297C         | ↵      |
| lfloor;           | U+0230A         | ⌊      |
| Lfr;              | U+1D50F         | ℓ      |
| lfr;              | U+1D529         | ℓ      |
| lg;               | U+02276         | ⩵      |
| lgE;              | U+02A91         | ↯      |
| lHar;             | U+02962         | ↵      |
| lhard;            | U+021BD         | ↵      |
| lharu;            | U+021BC         | ←      |
| lharul;           | U+0296A         | ↵      |

| Nom                  | Personnages) | Glyphe |
|----------------------|--------------|--------|
| lhblk;               | U+02584      | ■      |
| LJcy;                | U+00409      | é      |
| ljcy;                | U+00459      | љ      |
| Ll;                  | U+022D8      | ≪≪     |
| ll;                  | U+0226A      | ≪      |
| llarr;               | U+021C7      | ⇐      |
| llcorner;            | U+0231E      | └      |
| Lleftarrow;          | U+021DA      | ⇐      |
| llhard;              | U+0296B      | ⇐      |
| lltri;               | U+025FA      | ▷      |
| Lmidot;              | U+0013F      | je     |
| lmidot;              | U+00140      | ı      |
| lmoust;              | U+023B0      | ┐      |
| lmoustache;          | U+023B0      | ┐      |
| lnap;                | U+02A89      | ≈      |
| lnapprox;            | U+02A89      | ≈      |
| lnE;                 | U+02268      | ≇      |
| lne;                 | U+02A87      | ≇      |
| lneq;                | U+02A87      | ≇      |
| lneqq;               | U+02268      | ≇      |
| lnsim;               | U+022E6      | ≈      |
| loang;               | U+027EC      | ⌋      |
| loarr;               | U+021FD      | ←      |
| lobrk;               | U+027E6      | ⌋      |
| LongLeftArrow;       | U+027F5      | ←      |
| Longleftarrow;       | U+027F8      | ⇐      |
| longleftarrow;       | U+027F5      | ←      |
| LongLeftRightArrow;  | U+027F7      | ↔      |
| Longlefttrightarrow; | U+027FA      | ⇔      |
| longlefttrightarrow; | U+027F7      | ↔      |
| longmapsto;          | U+027FC      | ↦      |
| LongRightArrow;      | U+027F6      | →      |
| Longrightarrow;      | U+027F9      | ⇒      |
| longrightarrow;      | U+027F6      | →      |
| looparrowleft;       | U+021AB      | ↶      |
| looparrowright;      | U+021AC      | ↷      |

| Nom              | Personnages) | Glyphe |
|------------------|--------------|--------|
| lopar;           | U+02985      | ℔      |
| Lopf;            | U+1D543      | ℔      |
| lopf;            | U+1D55D      | ℔      |
| loplus;          | U+02A2D      | ℔      |
| lotimes;         | U+02A34      | ℔      |
| lowast;          | U+02217      | *      |
| lowbar;          | U+0005F      | —      |
| LowerLeftArrow;  | U+02199      | ↙      |
| LowerRightArrow; | U+02198      | ↘      |
| loz;             | U+025CA      | ◇      |
| lozenge;         | U+025CA      | ◇      |
| lozf;            | U+029EB      | ◆      |
| lpar;            | U+00028      | (      |
| lparlt;          | U+02993      | ⤵      |
| lrarr;           | U+021C6      | ↔      |
| lrcorner;        | U+0231F      | ┐      |
| lrhar;           | U+021CB      | ↔      |
| lrhard;          | U+0296D      | ⇒      |
| lrm;             | U+0200E      |        |
| lrtri;           | U+022BF      | ⚡      |
| lsaquo;          | U+02039      | ‹      |
| Lscr;            | U+02112      | ℒ      |
| lscr;            | U+1D4C1      | ℓ      |
| Lsh;             | U+021B0      | ↶      |
| lsh;             | U+021B0      | ↶      |
| lsim;            | U+02272      | ≈      |
| lsime;           | U+02A8D      | ≈      |
| lsimg;           | U+02A8F      | ≈      |
| lsqb;            | U+0005B      | [      |
| lsquo;           | U+02018      | '      |
| lsquor;          | U+0201A      | ,      |
| Lstrok;          | U+00141      | Ł      |
| lstrok;          | U+00142      | ł      |
| LT;              | U+0003C      | <      |
| LT               | U+0003C      | <      |
| Lt;              | U+0226A      | <<     |



| Nom            | Personnages)    | Glyphe |
|----------------|-----------------|--------|
| lt;            | U+0003C         | <      |
| lt             | U+0003C         | <      |
| ltcc;          | U+02AA6         | ◁      |
| ltcir;         | U+02A79         | ◁̂     |
| ltdot;         | U+022D6         | ◁̇     |
| lthree;        | U+022CB         | λ      |
| ltimes;        | U+022C9         | ⋈      |
| ltlarr;        | U+02976         | ⌞      |
| ltquest;       | U+02A7B         | ⌚      |
| ltri;          | U+025C3         | ◁      |
| ltrie;         | U+022B4         | ⊴      |
| ltrif;         | U+025C2         | ◁      |
| ltrPar;        | U+02996         | ⌞̂     |
| lurdshar;      | U+0294A         | ↵      |
| luruhar;       | U+02966         | ↵̂     |
| lvertneqq;     | U+02268 U+0FE00 | ≠̸     |
| lvnE;          | U+02268 U+0FE00 | ≠̸     |
| macr;          | U+000AF         | -      |
| macr           | U+000AF         | -      |
| male;          | U+02642         | ♂      |
| malt;          | U+02720         | ⌘      |
| maltese;       | U+02720         | ⌘      |
| Map;           | U+02905         | ↦⇒     |
| map;           | U+021A6         | ↦      |
| mapsto;        | U+021A6         | ↦      |
| mapstodown;    | U+021A7         | ↴      |
| mapstoleft;    | U+021A4         | ↵      |
| mapstoup;      | U+021A5         | ↶      |
| marker;        | U+025AE         | ■      |
| mcomma;        | U+02A29         | ⌞̂     |
| Mcy;           | U+0041C         | Ì      |
| mcy;           | U+0043C         | М      |
| mdash;         | U+02014         | —      |
| mDDot;         | U+0223A         | ⋮      |
| measuredangle; | U+02221         | ∠      |
| MediumSpace;   | U+0205F         |        |

| Nom        | Personnages)    | Glyphe         |
|------------|-----------------|----------------|
| Mellintrf; | U+02133         | $\mathcal{M}$  |
| Mfr;       | U+1D510         | $\mathfrak{M}$ |
| mfr;       | U+1D52A         | $\mathfrak{m}$ |
| mho;       | U+02127         | $\mathfrak{O}$ |
| micro;     | U+000B5         | $\mu$          |
| micro      | U+000B5         | $\mu$          |
| mid;       | U+02223         |                |
| midast;    | U+0002A         | *              |
| midcir;    | U+02AF0         | ◊              |
| middot;    | U+000B7         | .              |
| middot     | U+000B7         | .              |
| minus;     | U+02212         | −              |
| minusb;    | U+0229F         | ⊖              |
| minusd;    | U+02238         | ÷              |
| minusdu;   | U+02A2A         | ⋮              |
| MinusPlus; | U+02213         | ±              |
| mlcp;      | U+02ADB         | ⋈              |
| mldr;      | U+02026         | ...            |
| mnplus;    | U+02213         | ±              |
| models;    | U+022A7         | ⋈              |
| Mopf;      | U+1D544         | $\mathbb{M}$   |
| mopf;      | U+1D55E         | $\mathfrak{m}$ |
| mp;        | U+02213         | ±              |
| Mscr;      | U+02133         | $\mathcal{M}$  |
| mscr;      | U+1D4C2         | $\mathfrak{m}$ |
| mstpos;    | U+0223E         | ∞              |
| Mu;        | U+0039C         | Μ              |
| mu;        | U+003BC         | μ              |
| multimap;  | U+022B8         | ↦              |
| mumap;     | U+022B8         | ↦              |
| nabla;     | U+02207         | ∇              |
| Nacute;    | U+00143         | Ń              |
| nacute;    | U+00144         | ń              |
| nang;      | U+02220 U+020D2 | ∠              |
| nap;       | U+02249         | ≈              |
| napE;      | U+02A70 U+00338 | ≅              |

| Nom                    | Personnages)    | Glyphe |
|------------------------|-----------------|--------|
| napid;                 | U+0224B U+00338 | ≈/     |
| napos;                 | U+00149         | ’n     |
| napprox;               | U+02249         | ≈      |
| natur;                 | U+0266E         | ₥      |
| natural;               | U+0266E         | ₥      |
| naturals;              | U+02115         | N      |
| nbsp;                  | U+000A0         |        |
| nbspc                  | U+000A0         |        |
| nbump;                 | U+0224E U+00338 | ≈/     |
| nbumpe;                | U+0224F U+00338 | ≈/     |
| ncap;                  | U+02A43         | ᮑ      |
| Ncaron;                | U+00147         | Ň      |
| ncaron;                | U+00148         | ø      |
| Ncedil;                | U+00145         | Ŋ      |
| ncedil;                | U+00146         | ñ      |
| ncong;                 | U+02247         | ≇      |
| ncongdot;              | U+02A6D U+00338 | ≅/     |
| ncup;                  | U+02A42         | ᮒ      |
| Ncy;                   | U+0041D         | Ѐ      |
| ncy;                   | U+0043D         | і      |
| ndash;                 | U+02013         | –      |
| ne;                    | U+02260         | ≠      |
| nearhk;                | U+02924         | ↯      |
| neArr;                 | U+021D7         | ↗      |
| nearr;                 | U+02197         | ↗      |
| nearrow;               | U+02197         | ↗      |
| nedot;                 | U+02250 U+00338 | ≐/     |
| NegativeMediumSpace;   | U+0200B         |        |
| NegativeThickSpace;    | U+0200B         |        |
| NegativeThinSpace;     | U+0200B         |        |
| NegativeVeryThinSpace; | U+0200B         |        |
| nequiv;                | U+02262         | ≡      |
| nesear;                | U+02928         | ↷      |
| nesim;                 | U+02242 U+00338 | ≈/     |
| NestedGreaterGreater;  | U+0226B         | ≫      |
| NestedLessLess;        | U+0226A         | ≪      |

| Nom              | Personnages)    | Glyphe |
|------------------|-----------------|--------|
| NewLine;         | U+0000A         | □      |
| nexist;          | U+02204         | 𐀔      |
| nexists;         | U+02204         | 𐀔      |
| Nfr;             | U+1D511         | ℵ      |
| nfr;             | U+1D52B         | ℵ      |
| ngE;             | U+02267 U+00338 | ≧/     |
| nge;             | U+02271         | ≧      |
| ngeq;            | U+02271         | ≧      |
| ngeqq;           | U+02267 U+00338 | ≧/     |
| ngeqslant;       | U+02A7E U+00338 | ≧/     |
| nges;            | U+02A7E U+00338 | ≧/     |
| nGg;             | U+022D9 U+00338 | ≫/     |
| ngsim;           | U+02275         | ≧      |
| nGt;             | U+0226B U+020D2 | ≫      |
| ngt;             | U+0226F         | ➤      |
| ngtr;            | U+0226F         | ➤      |
| nGtv;            | U+0226B U+00338 | ≫/     |
| nhArr;           | U+021CE         | ↔      |
| nharr;           | U+021AE         | ↔      |
| nhpar;           | U+02AF2         | ‡      |
| ni;              | U+0220B         | ∋      |
| nis;             | U+022FC         | ∋      |
| nisd;            | U+022FA         | ∋      |
| niv;             | U+0220B         | ∋      |
| NJcy;            | U+0040A         | Ѓ      |
| njcy;            | U+0045A         | Ѓ      |
| nlArr;           | U+021CD         | ↔      |
| nlarr;           | U+0219A         | ↔      |
| nldr;            | U+02025         | ..     |
| nlE;             | U+02266 U+00338 | ≦/     |
| nle;             | U+02270         | ≦      |
| nLeftarrow;      | U+021CD         | ↔      |
| nleftarrow;      | U+0219A         | ↔      |
| nLeftrightarrow; | U+021CE         | ↔      |
| nleftrightarrow; | U+021AE         | ↔      |
| nleq;            | U+02270         | ≦      |

| Nom                   | Personnages)    | Glyphe |
|-----------------------|-----------------|--------|
| nleqq;                | U+02266 U+00338 | ≧      |
| nleqslant;            | U+02A7D U+00338 | ≧      |
| nles;                 | U+02A7D U+00338 | ≧      |
| nless;                | U+0226E         | ⋈      |
| nLl;                  | U+022D8 U+00338 | ≧      |
| nlsim;                | U+02274         | ⋈      |
| nLt;                  | U+0226A U+020D2 | ≧      |
| nlt;                  | U+0226E         | ⋈      |
| nltri;                | U+022EA         | ⋈      |
| nltrie;               | U+022CE         | ⋈      |
| nLtv;                 | U+0226A U+00338 | ≧      |
| nmid;                 | U+02224         | †      |
| NoBreak;              | U+02060         |        |
| NonBreakingSpace;     | U+000A0         |        |
| Nopf;                 | U+02115         | N      |
| nopf;                 | U+1D55F         | ℳ      |
| Not;                  | U+02AEC         | ⊃      |
| not;                  | U+000AC         | ¬      |
| not                   | U+000AC         | ¬      |
| NotCongruent;         | U+02262         | ≇      |
| NotCupCap;            | U+0226D         | ⋈      |
| NotDoubleVerticalBar; | U+02226         | ⋈      |
| NotElement;           | U+02209         | ∉      |
| NotEqual;             | U+02260         | ≠      |
| NotEqualTilde;        | U+02242 U+00338 | ≇      |
| NotExists;            | U+02204         | ∄      |
| NotGreater;           | U+0226F         | ⋈      |
| NotGreaterEqual;      | U+02271         | ⋈      |
| NotGreaterFullEqual;  | U+02267 U+00338 | ≧      |
| NotGreaterGreater;    | U+0226B U+00338 | ≫      |
| NotGreaterLess;       | U+02279         | ⋈      |
| NotGreaterSlantEqual; | U+02A7E U+00338 | ≧      |
| NotGreaterTilde;      | U+02275         | ⋈      |
| NotHumpDownHump;      | U+0224E U+00338 | ≇      |
| NotHumpEqual;         | U+0224F U+00338 | ≇      |
| notin;                | U+02209         | ∉      |

| Nom                      | Personnages)    | Glyphe |
|--------------------------|-----------------|--------|
| notindot;                | U+022F5 U+00338 | ∉      |
| notinE;                  | U+022F9 U+00338 | ∕      |
| notinva;                 | U+02209         | ∉      |
| notinvb;                 | U+022F7         | ∕      |
| notinvc;                 | U+022F6         | ∕      |
| NotLeftTriangle;         | U+022EA         | ⋈      |
| NotLeftTriangleBar;      | U+029CF U+00338 | ⋈      |
| NotLeftTriangleEqual;    | U+022CE         | ⋈      |
| NotLess;                 | U+0226E         | ⋈      |
| NotLessEqual;            | U+02270         | ⋈      |
| NotLessGreater;          | U+02278         | ⋈      |
| NotLessLess;             | U+0226A U+00338 | ⋈      |
| NotLessSlantEqual;       | U+02A7D U+00338 | ⋈      |
| NotLessTilde;            | U+02274         | ⋈      |
| NotNestedGreaterGreater; | U+02AA2 U+00338 | ⋈      |
| NotNestedLessLess;       | U+02AA1 U+00338 | ⋈      |
| notni;                   | U+0220C         | ⋈      |
| notniva;                 | U+0220C         | ⋈      |
| notnivb;                 | U+022FE         | ⋈      |
| notnivc;                 | U+022FD         | ⋈      |
| NotPrecedes;             | U+02280         | ⋈      |
| NotPrecedesEqual;        | U+02AAF U+00338 | ⋈      |
| NotPrecedesSlantEqual;   | U+022E0         | ⋈      |
| NotReverseElement;       | U+0220C         | ⋈      |
| NotRightTriangle;        | U+022EB         | ⋈      |
| NotRightTriangleBar;     | U+029D0 U+00338 | ⋈      |
| NotRightTriangleEqual;   | U+022ED         | ⋈      |
| NotSquareSubset;         | U+0228F U+00338 | ⋈      |
| NotSquareSubsetEqual;    | U+022E2         | ⋈      |
| NotSquareSuperset;       | U+02290 U+00338 | ⋈      |
| NotSquareSupersetEqual;  | U+022E3         | ⋈      |
| NotSubset;               | U+02282 U+020D2 | ⋈      |
| NotSubsetEqual;          | U+02288         | ⋈      |
| NotSucceeds;             | U+02281         | ⋈      |
| NotSucceedsEqual;        | U+02AB0 U+00338 | ⋈      |
| NotSucceedsSlantEqual;   | U+022E1         | ⋈      |

| Nom                | Personnages)    | Glyphe |
|--------------------|-----------------|--------|
| NotSucceedsTilde;  | U+0227F U+00338 | ≧/     |
| NotSuperset;       | U+02283 U+020D2 | ⊃      |
| NotSupersetEqual;  | U+02289         | ⊈      |
| NotTilde;          | U+02241         | ↯      |
| NotTildeEqual;     | U+02244         | ≠      |
| NotTildeFullEqual; | U+02247         | ≠̸     |
| NotTildeTilde;     | U+02249         | ≠̃     |
| NotVerticalBar;    | U+02224         | †      |
| npar;              | U+02226         | ‖      |
| nparallel;         | U+02226         | ‖      |
| nparsl;            | U+02AFD U+020E5 | ℵ      |
| npart;             | U+02202 U+00338 | ∂      |
| npolint;           | U+02A14         | ∫      |
| npr;               | U+02280         | ↯      |
| nprcue;            | U+022E0         | ↯̸     |
| npre;              | U+02AAF U+00338 | ≧/     |
| nprec;             | U+02280         | ↯      |
| npreceq;           | U+02AAF U+00338 | ≧/     |
| nrArr;             | U+021CF         | ⇈      |
| nrarr;             | U+0219B         | ↗      |
| nrarrc;            | U+02933 U+00338 | ↗̸     |
| nrarrw;            | U+0219D U+00338 | ↗̸     |
| nRightarrow;       | U+021CF         | ⇈      |
| nrightarrow;       | U+0219B         | ↗      |
| nrtri;             | U+022EB         | ▹      |
| nrtrie;            | U+022ED         | ▹̸     |
| nsc;               | U+02281         | ↯̸     |
| nsccue;            | U+022E1         | ↯̸̸    |
| nsce;              | U+02AB0 U+00338 | ≧/     |
| Nscr;              | U+1D4A9         | ℒ      |
| nscr;              | U+1D4C3         | ℓ      |
| nshortmid;         | U+02224         | †      |
| nshortparallel;    | U+02226         | ‖      |
| nsim;              | U+02241         | ↯      |
| nsime;             | U+02244         | ≠      |
| nsimeq;            | U+02244         | ≠      |

| Nom               | Personnages)    | Glyphe |
|-------------------|-----------------|--------|
| nsmid;            | U+02224         | †      |
| nspar;            | U+02226         | ‡      |
| nsqsube;          | U+022E2         | ≧̸     |
| nsqsupe;          | U+022E3         | ≨̸     |
| nsub;             | U+02284         | ¢      |
| nsubE;            | U+02AC5 U+00338 | ≧̸     |
| nsube;            | U+02288         | ¢      |
| nsubset;          | U+02282 U+020D2 | ⊂      |
| nsubseteq;        | U+02288         | ¢      |
| nsubseteqq;       | U+02AC5 U+00338 | ≧̸     |
| nsucc;            | U+02281         | ✕      |
| nsucceq;          | U+02AB0 U+00338 | ≧̸     |
| nsup;             | U+02285         | ¢      |
| nsupE;            | U+02AC6 U+00338 | ≧̸     |
| nsupe;            | U+02289         | ¢      |
| nsupset;          | U+02283 U+020D2 | ⊃      |
| nsupseteq;        | U+02289         | ¢      |
| nsupseteqq;       | U+02AC6 U+00338 | ≧̸     |
| ntgl;             | U+02279         | ₧      |
| Ntilde;           | U+000D1         | Ñ      |
| Ntilde            | U+000D1         | Ñ      |
| ntilde;           | U+000F1         | ñ      |
| ntilde            | U+000F1         | ñ      |
| ntl;              | U+02278         | ₧      |
| ntriangleleft;    | U+022EA         | ⋈      |
| ntrianglelefteq;  | U+022CE         | ⋈      |
| ntriangleright;   | U+022EB         | ⋈      |
| ntrianglerighteq; | U+022ED         | ⋈      |
| Nu;               | U+0039D         | Ν      |
| nu;               | U+003BD         | ν      |
| num;              | U+00023         | #      |
| numero;           | U+02116         | №      |
| numsp;            | U+02007         |        |
| nvap;             | U+0224D U+020D2 | ≈      |
| nVDash;           | U+022AF         | ‖      |
| nVdash;           | U+022AE         | ‖      |



| Nom      | Personnages)    | Glyphe |
|----------|-----------------|--------|
| nvDash;  | U+022AD         | ≠      |
| nvdash;  | U+022AC         | ⋮      |
| nvge;    | U+02265 U+020D2 | ≥      |
| nvgt;    | U+0003E U+020D2 | >      |
| nvHarr;  | U+02904         | ↔      |
| nvinfin; | U+029DE         | ♾      |
| nvlArr;  | U+02902         | ⇐      |
| nvle;    | U+02264 U+020D2 | ≤      |
| nvlt;    | U+0003C U+020D2 | <      |
| nvltrle; | U+022B4 U+020D2 | ⩽      |
| nvrArr;  | U+02903         | ⇒      |
| nvrtrie; | U+022B5 U+020D2 | ⩾      |
| nvsim;   | U+0223C U+020D2 | ~      |
| nwarhk;  | U+02923         | ↯      |
| nwArr;   | U+021D6         | ↗      |
| nwarr;   | U+02196         | ↖      |
| nwarrow; | U+02196         | ↖      |
| nwnear;  | U+02927         | ↘      |
| Oacute;  | U+000D3         | Ó      |
| Oacute   | U+000D3         | Ó      |
| oacute;  | U+000F3         | ó      |
| oacute   | U+000F3         | ó      |
| oast;    | U+0229B         | ⊗      |
| ocir;    | U+0229A         | ⊙      |
| Ocirc;   | U+000D4         | Ô      |
| Ocirc    | U+000D4         | Ô      |
| ocirc;   | U+000F4         | ô      |
| ocirc    | U+000F4         | ô      |
| Ocy;     | U+0041E         | О      |
| ocy;     | U+0043E         | о      |
| odash;   | U+0229D         | ⊖      |
| Odblac;  | U+00150         | À      |
| odblac;  | U+00151         | ă      |
| odiv;    | U+02A38         | ⊕      |
| odot;    | U+02299         | ⊙      |
| odsold;  | U+029BC         | ⊗      |

| Nom                   | Personnages) | Glyphe |
|-----------------------|--------------|--------|
| OElig;                | U+00152      | Œ      |
| oelig;                | U+00153      | œ      |
| ofcir;                | U+029BF      | ⦿      |
| Ofr;                  | U+1D512      | Ɔ      |
| ofr;                  | U+1D52C      | ɐ      |
| ogon;                 | U+002DB      | ˆ      |
| Ograve;               | U+000D2      | Ò      |
| Ograve                | U+000D2      | Ò      |
| ograve;               | U+000F2      | ò      |
| ograve                | U+000F2      | ò      |
| ogt;                  | U+029C1      | ⊗      |
| ohbar;                | U+029B5      | ⊖      |
| ohm;                  | U+003A9      | Ω      |
| oint;                 | U+0222E      | ℳ      |
| olarr;                | U+021BA      | ↷      |
| olcir;                | U+029BE      | ⊙      |
| olcross;              | U+029BB      | ⊗      |
| oline;                | U+0203E      | -      |
| olt;                  | U+029C0      | ⊗      |
| Omacr;                | U+0014C      | Ō      |
| omacr;                | U+0014D      | ō      |
| Omega;                | U+003A9      | Ω      |
| omega;                | U+003C9      | ω      |
| Omicron;              | U+0039F      | Ο      |
| omicron;              | U+003BF      | ο      |
| omid;                 | U+029B6      | ⊕      |
| ominus;               | U+02296      | ⊖      |
| Oopf;                 | U+1D546      | ⓪      |
| oopf;                 | U+1D560      | ⓪      |
| opar;                 | U+029B7      | ⓪      |
| OpenCurlyDoubleQuote; | U+0201C      | "      |
| OpenCurlyQuote;       | U+02018      | '      |
| operp;                | U+029B9      | ⊕      |
| oplus;                | U+02295      | ⊕      |
| Or;                   | U+02A54      | ℷ      |
| or;                   | U+02228      | ∨      |

| Nom              | Personnages) | Glyphe |
|------------------|--------------|--------|
| orarr;           | U+021BB      | ↻      |
| ord;             | U+02A5D      | ∀      |
| order;           | U+02134      | ℔      |
| orderof;         | U+02134      | ℔      |
| ordf;            | U+000AA      | Ɽ      |
| ordf             | U+000AA      | Ɽ      |
| ordm;            | U+000BA      | Ɱ      |
| ordm             | U+000BA      | Ɱ      |
| origof;          | U+022B6      | ↻↻     |
| oror;            | U+02A56      | ℔      |
| orslope;         | U+02A57      | ℔      |
| orv;             | U+02A5B      | ℔      |
| oS;              | U+024C8      | Ⓢ      |
| Oscr;            | U+1D4AA      | ℴ      |
| oscr;            | U+02134      | ℔      |
| Oslash;          | U+000D8      | Ø      |
| Oslash           | U+000D8      | Ø      |
| oslash;          | U+000F8      | ø      |
| oslash           | U+000F8      | ø      |
| osol;            | U+02298      | ⊘      |
| Otilde;          | U+000D5      | Õ      |
| Otilde           | U+000D5      | Õ      |
| otilde;          | U+000F5      | õ      |
| otilde           | U+000F5      | õ      |
| Otimes;          | U+02A37      | ⊗      |
| otimes;          | U+02297      | ⊗      |
| otimesas;        | U+02A36      | ⊗      |
| Ouml;            | U+000D6      | Ö      |
| Ouml             | U+000D6      | Ö      |
| ouml;            | U+000F6      | ö      |
| ouml             | U+000F6      | ö      |
| ovbar;           | U+0233D      | ⏞      |
| OverBar;         | U+0203E      | ¯      |
| OverBrace;       | U+023DE      | ⏞      |
| OverBracket;     | U+023B4      | ⏞      |
| OverParenthesis; | U+023DC      | ⏞      |

| Nom        | Personnages) | Glyphe |
|------------|--------------|--------|
| par;       | U+02225      | ∥      |
| para;      | U+000B6      | ¶      |
| para       | U+000B6      | ¶      |
| parallel;  | U+02225      | ∥      |
| parsim;    | U+02AF3      | ‡      |
| parsl;     | U+02AFD      | ∥      |
| part;      | U+02202      | ∂      |
| PartialD;  | U+02202      | ∂      |
| Pcy;       | U+0041F      | Π      |
| pcy;       | U+0043F      | π      |
| percnt;    | U+00025      | %      |
| period;    | U+0002E      | .      |
| permil;    | U+02030      | ‰      |
| perp;      | U+022A5      | ⊥      |
| pertenk;   | U+02031      | ‱      |
| Pfr;       | U+1D513      | ℔      |
| pfr;       | U+1D52D      | ℥      |
| Phi;       | U+003A6      | Φ      |
| phi;       | U+003C6      | φ      |
| phiv;      | U+003D5      | ϕ      |
| phmmat;    | U+02133      | ℳ      |
| phone;     | U+0260E      | ☎      |
| Pi;        | U+003A0      | Π      |
| pi;        | U+003C0      | π      |
| pitchfork; | U+022D4      | ‡      |
| piv;       | U+003D6      | ϖ      |
| planck;    | U+0210F      | ℏ      |
| planckh;   | U+0210E      | ℏ      |
| plankv;    | U+0210F      | ℏ      |
| plus;      | U+0002B      | +      |
| plusacir;  | U+02A23      | ⋈      |
| plusb;     | U+0229E      | ⊞      |
| pluscir;   | U+02A22      | ⋇      |
| plusdo;    | U+02214      | ⋉      |
| plusdu;    | U+02A25      | ⋊      |
| pluse;     | U+02A72      | ±      |

| Nom                 | Personnages) | Glyphe |
|---------------------|--------------|--------|
| PlusMinus;          | U+000B1      | ±      |
| plusmn;             | U+000B1      | ±      |
| plusmn              | U+000B1      | ±      |
| plussim;            | U+02A26      | ±̸     |
| plustwo;            | U+02A27      | ±₂     |
| pm;                 | U+000B1      | ±      |
| Poincareplane;      | U+0210C      | ℋ      |
| pointint;           | U+02A15      | ℳ      |
| Popf;               | U+02119      | ℙ      |
| popf;               | U+1D561      | ℙ      |
| pound;              | U+000A3      | £      |
| pound               | U+000A3      | £      |
| Pr;                 | U+02ABB      | ↵      |
| pr;                 | U+0227A      | <      |
| prap;               | U+02AB7      | ≈      |
| prcue;              | U+0227C      | ≲      |
| prE;                | U+02AB3      | ≳      |
| pre;                | U+02AAF      | ≲      |
| prec;               | U+0227A      | <      |
| precapprox;         | U+02AB7      | ≈      |
| preccurlyeq;        | U+0227C      | ≲      |
| Precedes;           | U+0227A      | <      |
| PrecedesEqual;      | U+02AAF      | ≲      |
| PrecedesSlantEqual; | U+0227C      | ≲      |
| PrecedesTilde;      | U+0227E      | ≲̃     |
| preceq;             | U+02AAF      | ≲      |
| precnapprox;        | U+02AB9      | ≈̃     |
| precneqq;           | U+02AB5      | ≇      |
| precnsim;           | U+022E8      | ≲̂     |
| precsim;            | U+0227E      | ≲̃     |
| Prime;              | U+02033      | "      |
| prime;              | U+02032      | '      |
| primes;             | U+02119      | ℙ      |
| prnap;              | U+02AB9      | ≈̃     |
| prnE;               | U+02AB5      | ≇      |
| prnsim;             | U+022E8      | ≲̂     |

| Nom           | Personnages)    | Glyphe   |
|---------------|-----------------|----------|
| prod;         | U+0220F         | ∏        |
| Product;      | U+0220F         | ∏        |
| profalar;     | U+0232E         | ◻        |
| proflin;      | U+02312         | ∩        |
| profsurf;     | U+02313         | ∪        |
| prop;         | U+0221D         | α        |
| Proportion;   | U+02237         | ::       |
| Proportional; | U+0221D         | α        |
| propto;       | U+0221D         | α        |
| prsim;        | U+0227E         | ≲        |
| prurel;       | U+022B0         | ↯        |
| Pscr;         | U+1D4AB         | <i>ℙ</i> |
| pscr;         | U+1D4C5         | <i>℘</i> |
| Psi;          | U+003A8         | Ψ        |
| psi;          | U+003C8         | ψ        |
| puncsp;       | U+02008         |          |
| Qfr;          | U+1D514         | ℚ        |
| qfr;          | U+1D52E         | q        |
| qint;         | U+02A0C         | ∫∫∫∫     |
| Qopf;         | U+0211A         | ℚ        |
| qopf;         | U+1D562         | ℚ        |
| qprime;       | U+02057         | '''      |
| Qscr;         | U+1D4AC         | <i>ℚ</i> |
| qscr;         | U+1D4C6         | <i>q</i> |
| quaternions;  | U+0210D         | ℍ        |
| quatint;      | U+02A16         | ∫        |
| quest;        | U+0003F         | ?        |
| questeq;      | U+0225F         | ≐        |
| QUOT;         | U+00022         | "        |
| QUOT          | U+00022         | "        |
| quot;         | U+00022         | "        |
| quot          | U+00022         | "        |
| rAarr;        | U+021DB         | ⇒        |
| race;         | U+0223D U+00331 | ≈        |
| Racute;       | U+00154         | Ŕ        |
| racute;       | U+00155         | à        |

| Nom       | Personnages) | Glyphe |
|-----------|--------------|--------|
| radic;    | U+0221A      | √      |
| raemptyv; | U+029B3      | ∅      |
| Rang;     | U+027EB      | »      |
| rang;     | U+027E9      | }      |
| rangd;    | U+02992      | }      |
| range;    | U+029A5      | ≧      |
| rangle;   | U+027E9      | }      |
| raquo;    | U+000BB      | »      |
| raquo     | U+000BB      | »      |
| Rarr;     | U+021A0      | →      |
| rArr;     | U+021D2      | ⇒      |
| rarr;     | U+02192      | →      |
| rarrap;   | U+02975      | ⇒      |
| rarrb;    | U+021E5      | →      |
| rarrbfs;  | U+02920      | ↔      |
| rarrc;    | U+02933      | ↪      |
| rarrfs;   | U+0291E      | ↔      |
| rarrhk;   | U+021AA      | ↪      |
| rarrlp;   | U+021AC      | ↪      |
| rarrpl;   | U+02945      | ↪      |
| rarrsim;  | U+02974      | ⇒      |
| Rarrtl;   | U+02916      | ↪      |
| rarrtl;   | U+021A3      | ↪      |
| rarrw;    | U+0219D      | ↪      |
| rAtail;   | U+0291C      | ↪      |
| ratail;   | U+0291A      | ↪      |
| ratio;    | U+02236      | :      |
| rational; | U+0211A      | ℚ      |
| RBarr;    | U+02910      | ↪      |
| rBarr;    | U+0290F      | ↪      |
| rbarr;    | U+0290D      | ↪      |
| rbbrk;    | U+02773      | )      |
| rbrace;   | U+0007D      | }      |
| rbrack;   | U+0005D      | ]      |
| rbrke;    | U+0298C      | ]      |
| rbrksld;  | U+0298E      | ]      |

| Nom                   | Personnages) | Glyphe |
|-----------------------|--------------|--------|
| rbrkslu;              | U+02990      | ⌋      |
| Rcaron;               | U+00158      | Ø      |
| rcaron;               | U+00159      | ø      |
| Rcedil;               | U+00156      | Ŕ      |
| rcedil;               | U+00157      | ŕ      |
| rceil;                | U+02309      | ⌋      |
| rcub;                 | U+0007D      | }      |
| Rcy;                  | U+00420      | Р      |
| rcy;                  | U+00440      | Ѡ      |
| rdca;                 | U+02937      | ↳      |
| rdldhar;              | U+02969      | ⇌      |
| rdquo;                | U+0201D      | ”      |
| rdquor;               | U+0201D      | ”      |
| rdsh;                 | U+021B3      | ↳      |
| Re;                   | U+0211C      | ℜ      |
| real;                 | U+0211C      | ℜ      |
| realine;              | U+0211B      | ℔      |
| realpart;             | U+0211C      | ℜ      |
| reals;                | U+0211D      | ℞      |
| rect;                 | U+025AD      | ◻      |
| REG;                  | U+000AE      | ®      |
| REG                   | U+000AE      | ®      |
| reg;                  | U+000AE      | ®      |
| reg                   | U+000AE      | ®      |
| ReverseElement;       | U+0220B      | ∋      |
| ReverseEquilibrium;   | U+021CB      | ⇌      |
| ReverseUpEquilibrium; | U+0296F      | ⌋      |
| rfisht;               | U+0297D      | ↗      |
| rfloor;               | U+0230B      | ⌋      |
| Rfr;                  | U+0211C      | ℜ      |
| rfr;                  | U+1D52F      | ℞      |
| rHar;                 | U+02964      | ⇒      |
| rhard;                | U+021C1      | →      |
| rharu;                | U+021C0      | ↗      |
| rharul;               | U+0296C      | ⇒      |
| Rho;                  | U+003A1      | ρ      |



| Nom                  | Personnages) | Glyphe |
|----------------------|--------------|--------|
| rho;                 | U+003C1      | ρ      |
| rhov;                | U+003F1      | ϱ      |
| RightAngleBracket;   | U+027E9      | ⌋      |
| RightArrow;          | U+02192      | →      |
| Rightarrow;          | U+021D2      | ⇒      |
| rightarrow;          | U+02192      | →      |
| RightArrowBar;       | U+021E5      | →̄     |
| RightArrowLeftArrow; | U+021C4      | ↔      |
| rightarrowtail;      | U+021A3      | ↗      |
| RightCeiling;        | U+02309      | ⌈      |
| RightDoubleBracket;  | U+027E7      | ⌋⌋     |
| RightDownTeeVector;  | U+0295D      | ⌴      |
| RightDownVector;     | U+021C2      | ⌵      |
| RightDownVectorBar;  | U+02955      | ⌵̄     |
| RightFloor;          | U+0230B      | ⌋      |
| rightharpoonup;      | U+021C1      | ↗      |
| rightharpoonup;      | U+021C0      | ↖      |
| rightleftarrows;     | U+021C4      | ↔      |
| rightleftharpoons;   | U+021CC      | ⇔      |
| rightrightarrow;     | U+021C9      | ⇒      |
| rightsquigarrow;     | U+0219D      | ↗      |
| RightTee;            | U+022A2      | ⊢      |
| RightTeeArrow;       | U+021A6      | ↗      |
| RightTeeVector;      | U+0295B      | ⌴      |
| rightthreetimes;     | U+022CC      | ⋈      |
| RightTriangle;       | U+022B3      | ▷      |
| RightTriangleBar;    | U+029D0      | ▷̄     |
| RightTriangleEqual;  | U+022B5      | ▷̸     |
| RightUpDownVector;   | U+0294F      | ⌵      |
| RightUpTeeVector;    | U+0295C      | ⌴      |
| RightUpVector;       | U+021BE      | ⌶      |
| RightUpVectorBar;    | U+02954      | ⌶̄     |
| RightVector;         | U+021C0      | ↖      |
| RightVectorBar;      | U+02953      | ↖̄     |
| ring;                | U+002DA      | °      |
| risingdotseq;        | U+02253      | ⋈      |

| Nom           | Personnages) | Glyphe               |
|---------------|--------------|----------------------|
| rlarr;        | U+021C4      | $\rightleftarrows$   |
| rlhar;        | U+021CC      | $\rightleftharpoons$ |
| rlm;          | U+0200F      |                      |
| rmoust;       | U+023B1      | $\}$                 |
| rmoustache;   | U+023B1      | $\}$                 |
| rnmid;        | U+02AEE      | $\dagger$            |
| roang;        | U+027ED      | $\parallel$          |
| roarr;        | U+021FE      | $\rightarrow$        |
| robrk;        | U+027E7      | $\parallel$          |
| ropar;        | U+02986      | $\rangle$            |
| Ropf;         | U+0211D      | $\mathbb{R}$         |
| ropf;         | U+1D563      | $\mathfrak{r}$       |
| roplus;       | U+02A2E      | $\oplus$             |
| rotimes;      | U+02A35      | $\otimes$            |
| RoundImplies; | U+02970      | $\Rightarrow$        |
| rpar;         | U+00029      | )                    |
| rpargt;       | U+02994      | $\rangle$            |
| rppolint;     | U+02A12      | $\int$               |
| rrarr;        | U+021C9      | $\Rightarrow$        |
| Rightarrow;   | U+021DB      | $\Rightarrow$        |
| rsaquo;       | U+0203A      | $\rangle$            |
| Rscr;         | U+0211B      | $\mathcal{R}$        |
| rscr;         | U+1D4C7      | $\mathfrak{r}$       |
| Rsh;          | U+021B1      | $\P$                 |
| rsh;          | U+021B1      | $\P$                 |
| rsqb;         | U+0005D      | ]                    |
| rsquo;        | U+02019      | '                    |
| rsquor;       | U+02019      | '                    |
| rthree;       | U+022CC      | $\prec$              |
| rtimes;       | U+022CA      | $\rtimes$            |
| rtri;         | U+025B9      | $\triangleright$     |
| rtrie;        | U+022B5      | $\triangleright$     |
| rtrif;        | U+025B8      | $\triangleright$     |
| rtriltri;     | U+029CE      | $\boxtriangleright$  |
| RuleDelayed;  | U+029F4      | $\Rightarrow$        |
| ruluhar;      | U+02968      | $\rightleftharpoons$ |

| Nom                    | Personnages) | Glyphe |
|------------------------|--------------|--------|
| <code>rx;</code>       | U+0211E      | ℞      |
| <code>Sacute;</code>   | U+0015A      | Ś      |
| <code>sacute;</code>   | U+0015B      | ś      |
| <code>sbquo;</code>    | U+0201A      | ‚      |
| <code>Sc;</code>       | U+02ABC      | ⤿      |
| <code>sc;</code>       | U+0227B      | ⋗      |
| <code>scap;</code>     | U+02AB8      | ⋗̂     |
| <code>Scaron;</code>   | U+00160      | Š      |
| <code>scaron;</code>   | U+00161      | š      |
| <code>sccue;</code>    | U+0227D      | ⋗̂     |
| <code>scE;</code>      | U+02AB4      | ⋗̂     |
| <code>sce;</code>      | U+02AB0      | ⋗̂     |
| <code>Scedil;</code>   | U+0015E      | Ş      |
| <code>scedil;</code>   | U+0015F      | ş      |
| <code>Scirc;</code>    | U+0015C      | Ŝ      |
| <code>scirc;</code>    | U+0015D      | ŝ      |
| <code>scnap;</code>    | U+02ABA      | ⋗̂     |
| <code>scnE;</code>     | U+02AB6      | ⋗̂     |
| <code>scnsim;</code>   | U+022E9      | ⋗̂     |
| <code>scpolint;</code> | U+02A13      | ſ̣     |
| <code>scsim;</code>    | U+0227F      | ⋗̂     |
| <code>Scy;</code>      | U+00421      | С      |
| <code>scy;</code>      | U+00441      | с      |
| <code>sdot;</code>     | U+022C5      | ⋅      |
| <code>sdotb;</code>    | U+022A1      | ◻̣     |
| <code>sdote;</code>    | U+02A66      | ⋈      |
| <code>searhk;</code>   | U+02925      | ↯      |
| <code>seArr;</code>    | U+021D8      | ↯      |
| <code>searr;</code>    | U+02198      | ↯      |
| <code>searrow;</code>  | U+02198      | ↯      |
| <code>sect;</code>     | U+000A7      | §      |
| <code>sect</code>      | U+000A7      | §      |
| <code>semi;</code>     | U+0003B      | ;      |
| <code>seswar;</code>   | U+02929      | ↯      |
| <code>setminus;</code> | U+02216      | \      |
| <code>setmn;</code>    | U+02216      | \      |

| Nom              | Personnages) | Glyphe |
|------------------|--------------|--------|
| sext;            | U+02736      | ✱      |
| Sfr;             | U+1D516      | ℑ      |
| sfr;             | U+1D530      | ℓ      |
| sfrown;          | U+02322      | ⌒      |
| sharp;           | U+0266F      | #      |
| SHCHcy;          | U+00429      | Ил     |
| shchcy;          | U+00449      | é      |
| SHcy;            | U+00428      | Ø      |
| shcy;            | U+00448      | ø      |
| ShortDownArrow;  | U+02193      | ↓      |
| ShortLeftArrow;  | U+02190      | ←      |
| shortmid;        | U+02223      |        |
| shortparallel;   | U+02225      |        |
| ShortRightArrow; | U+02192      | →      |
| ShortUpArrow;    | U+02191      | ↑      |
| shy;             | U+000AD      |        |
| shy              | U+000AD      |        |
| Sigma;           | U+003A3      | Σ      |
| sigma;           | U+003C3      | σ      |
| sigmaf;          | U+003C2      | ς      |
| sigmav;          | U+003C2      | ς      |
| sim;             | U+0223C      | ~      |
| simdot;          | U+02A6A      | ⋈      |
| sime;            | U+02243      | ≈      |
| simeq;           | U+02243      | ≈      |
| simg;            | U+02A9E      | ≳      |
| simgE;           | U+02AA0      | ≳      |
| siml;            | U+02A9D      | ≲      |
| simlE;           | U+02A9F      | ≲      |
| simne;           | U+02246      | ≇      |
| simplus;         | U+02A24      | ⋎      |
| simrarr;         | U+02972      | ⇒      |
| slarr;           | U+02190      | ←      |
| SmallCircle;     | U+02218      | ◦      |
| smallsetminus;   | U+02216      | \      |
| smashp;          | U+02A33      | ⋈      |

| Nom                 | Personnages)    | Glyphe |
|---------------------|-----------------|--------|
| smeparsl;           | U+029E4         | ≠      |
| smid;               | U+02223         |        |
| smile;              | U+02323         | ☺      |
| smt;                | U+02AAA         | ≤      |
| smte;               | U+02AAC         | ≤      |
| smtes;              | U+02AAC U+0FE00 | ≤      |
| SOFTcy;             | U+0042C         | Ü      |
| softcy;             | U+0044C         | ü      |
| sol;                | U+0002F         | /      |
| solb;               | U+029C4         | ∅      |
| solbar;             | U+0233F         | ≠      |
| Sopf;               | U+1D54A         | ℳ      |
| sopf;               | U+1D564         | ℳ      |
| spades;             | U+02660         | ♠      |
| spadesuit;          | U+02660         | ♠      |
| spar;               | U+02225         |        |
| sqcap;              | U+02293         | ⊏      |
| sqcaps;             | U+02293 U+0FE00 | ⊏      |
| sqcup;              | U+02294         | ⊔      |
| sqcups;             | U+02294 U+0FE00 | ⊔      |
| Sqrt;               | U+0221A         | √      |
| sqsub;              | U+0228F         | ⊂      |
| sqsube;             | U+02291         | ⊆      |
| sqsubset;           | U+0228F         | ⊂      |
| sqsubsetq;          | U+02291         | ⊆      |
| sqsup;              | U+02290         | ⊃      |
| sqsupe;             | U+02292         | ⊇      |
| sqsupset;           | U+02290         | ⊃      |
| sqsupsetq;          | U+02292         | ⊇      |
| squ;                | U+025A1         | □      |
| Square;             | U+025A1         | □      |
| square;             | U+025A1         | □      |
| SquareIntersection; | U+02293         | ⊏      |
| SquareSubset;       | U+0228F         | ⊂      |
| SquareSubsetEqual;  | U+02291         | ⊆      |
| SquareSuperset;     | U+02290         | ⊃      |

| Nom                  | Personnages) | Glyphe |
|----------------------|--------------|--------|
| SquareSupersetEqual; | U+02292      | ⊃      |
| SquareUnion;         | U+02294      | ⊍      |
| squarf;              | U+025AA      | ▪      |
| squf;                | U+025AA      | ▪      |
| srarr;               | U+02192      | →      |
| Sscr;                | U+1D4AE      | ℒ      |
| sscr;                | U+1D4C8      | ℓ      |
| ssetmn;              | U+02216      | \      |
| ssmile;              | U+02323      | ⤿      |
| sstarf;              | U+022C6      | ★      |
| Star;                | U+022C6      | ★      |
| star;                | U+02606      | ☆      |
| starf;               | U+02605      | ★      |
| straightepsilon;     | U+003F5      | €      |
| straightphi;         | U+003D5      | ϕ      |
| strns;               | U+000AF      | -      |
| Sub;                 | U+022D0      | ⊆      |
| sub;                 | U+02282      | ⊂      |
| subdot;              | U+02ABD      | ⊆̇     |
| subE;                | U+02AC5      | ⊆      |
| sube;                | U+02286      | ⊆      |
| subedot;             | U+02AC3      | ⊆̇     |
| submult;             | U+02AC1      | ⊆*     |
| subnE;               | U+02ACB      | ⊆      |
| subne;               | U+0228A      | ⊆      |
| subplus;             | U+02ABF      | ⊆+     |
| subrarr;             | U+02979      | ⊆→     |
| Subset;              | U+022D0      | ⊆      |
| subset;              | U+02282      | ⊂      |
| subseteq;            | U+02286      | ⊆      |
| subseteqq;           | U+02AC5      | ⊆      |
| SubsetEqual;         | U+02286      | ⊆      |
| subsetneq;           | U+0228A      | ⊆      |
| subsetneqq;          | U+02ACB      | ⊆      |
| subsim;              | U+02AC7      | ⊆      |

| Nom                 | Personnages) | Glyphe |
|---------------------|--------------|--------|
| subsub;             | U+02AD5      | ⋋      |
| subsup;             | U+02AD3      | ⋊      |
| succ;               | U+0227B      | ⋗      |
| succapprox;         | U+02AB8      | ⋗̃     |
| succcurlyeq;        | U+0227D      | ⋗̃     |
| Succeeds;           | U+0227B      | ⋗      |
| SucceedsEqual;      | U+02AB0      | ⋗̃     |
| SucceedsSlantEqual; | U+0227D      | ⋗̃     |
| SucceedsTilde;      | U+0227F      | ⋗̃     |
| succeq;             | U+02AB0      | ⋗̃     |
| succnapprox;        | U+02ABA      | ⋗̃     |
| succneqq;           | U+02AB6      | ⋗̃     |
| succnsim;           | U+022E9      | ⋗̃     |
| succsim;            | U+0227F      | ⋗̃     |
| SuchThat;           | U+0220B      | ⋗      |
| Sum;                | U+02211      | ∑      |
| sum;                | U+02211      | ∑      |
| sung;               | U+0266A      | ♪      |
| Sup;                | U+022D1      | ⋗      |
| sup;                | U+02283      | ⋗      |
| sup1;               | U+000B9      | ¹      |
| sup1                | U+000B9      | ¹      |
| sup2;               | U+000B2      | ²      |
| sup2                | U+000B2      | ²      |
| sup3;               | U+000B3      | ³      |
| sup3                | U+000B3      | ³      |
| supdot;             | U+02ABE      | ⋗̇     |
| supdsub;            | U+02AD8      | ⋗⋊     |
| supE;               | U+02AC6      | ⋗̃     |
| supe;               | U+02287      | ⋗̃     |
| supedot;            | U+02AC4      | ⋗̇     |
| Superset;           | U+02283      | ⋗      |
| SupersetEqual;      | U+02287      | ⋗̃     |
| suphsol;            | U+027C9      | ⋗/     |
| suphsub;            | U+02AD7      | ⋗⋊     |
| suplarr;            | U+0297B      | ⋗̃     |

| Nom         | Personnages) | Glyphe |
|-------------|--------------|--------|
| supmult;    | U+02AC2      | ↯      |
| supnE;      | U+02ACC      | ↯̸     |
| supne;      | U+0228B      | ↯̸     |
| supplus;    | U+02AC0      | ↯̸     |
| Supset;     | U+022D1      | ⊃      |
| supset;     | U+02283      | ⊃      |
| supseteq;   | U+02287      | ⊇      |
| supseteqq;  | U+02AC6      | ⊇      |
| supsetneq;  | U+0228B      | ↯̸     |
| supsetneqq; | U+02ACC      | ↯̸     |
| supsim;     | U+02AC8      | ≈      |
| supsub;     | U+02AD4      | ⊂      |
| supsup;     | U+02AD6      | ⊃      |
| swarhk;     | U+02926      | ↯̸     |
| swArr;      | U+021D9      | ↯̸     |
| swarr;      | U+02199      | ↯̸     |
| swarrow;    | U+02199      | ↯̸     |
| swnwar;     | U+0292A      | ↯̸     |
| szlig;      | U+000DF      | ß      |
| szlig       | U+000DF      | ß      |
| Tab;        | U+00009      | □      |
| target;     | U+02316      | ⊕      |
| Tau;        | U+003A4      | Τ      |
| tau;        | U+003C4      | τ      |
| tbrk;       | U+023B4      | ⌞      |
| Tcaron;     | U+00164      | ř      |
| tcaron;     | U+00165      | ř      |
| Tcedil;     | U+00162      | Ţ      |
| tcedil;     | U+00163      | ţ      |
| Tcy;        | U+00422      | Т      |
| tcy;        | U+00442      | т      |
| tdot;       | U+020DB      | ◌̣     |
| telrec;     | U+02315      | ⊖      |
| Tfr;        | U+1D517      | ℱ      |
| tfr;        | U+1D531      | ℱ      |
| there4;     | U+02234      | ∴      |



| Nom             | Personnages)    | Glyphe |
|-----------------|-----------------|--------|
| Therefore;      | U+02234         | ∴      |
| therefore;      | U+02234         | ∴      |
| Theta;          | U+00398         | Θ      |
| theta;          | U+003B8         | θ      |
| thetasym;       | U+003D1         | ϑ      |
| thetav;         | U+003D1         | ϑ      |
| thickapprox;    | U+02248         | ≈      |
| thicksim;       | U+0223C         | ∼      |
| ThickSpace;     | U+0205F U+0200A |        |
| thinsp;         | U+02009         |        |
| ThinSpace;      | U+02009         |        |
| thkap;          | U+02248         | ≈      |
| thksim;         | U+0223C         | ∼      |
| THORN;          | U+000DE         | E      |
| THORN           | U+000DE         | E      |
| thorn;          | U+000FE         | e      |
| thorn           | U+000FE         | e      |
| Tilde;          | U+0223C         | ∼      |
| tilde;          | U+002DC         | ˜      |
| TildeEqual;     | U+02243         | ≈      |
| TildeFullEqual; | U+02245         | ≐      |
| TildeTilde;     | U+02248         | ≈      |
| times;          | U+000D7         | ×      |
| times           | U+000D7         | ×      |
| timesb;         | U+022A0         | ⊗      |
| timesbar;       | U+02A31         | ⊠      |
| timesd;         | U+02A30         | ⊠      |
| tint;           | U+0222D         | ∫      |
| toea;           | U+02928         | ⌘      |
| top;            | U+022A4         | ⌞      |
| topbot;         | U+02336         | ⌏      |
| topcir;         | U+02AF1         | ⌠      |
| Topf;           | U+1D54B         | Ⓙ      |
| topf;           | U+1D565         | Ⓣ      |
| topfork;        | U+02ADA         | ⌢      |
| tosa;           | U+02929         | ⌘      |

| Nom                | Personnages) | Glyphe   |
|--------------------|--------------|----------|
| tprime;            | U+02034      | '''      |
| TRADE;             | U+02122      | ™        |
| trade;             | U+02122      | ™        |
| triangle;          | U+025B5      | △        |
| triangledown;      | U+025BF      | ▽        |
| triangleleft;      | U+025C3      | ◁        |
| trianglelefteq;    | U+022B4      | ⋖        |
| triangleq;         | U+0225C      | ≐        |
| triangleright;     | U+025B9      | ▷        |
| trianglerighteq;   | U+022B5      | ⋗        |
| tridot;            | U+025CE      | △        |
| trie;              | U+0225C      | ≐        |
| triminus;          | U+02A3A      | △        |
| TripleDot;         | U+020DB      | ⋯        |
| tripplus;          | U+02A39      | △        |
| trisb;             | U+029CD      | △        |
| tritime;           | U+02A3B      | △        |
| trpezium;          | U+023E2      | □        |
| Tscr;              | U+1D4AF      | <i>℣</i> |
| tscr;              | U+1D4C9      | <i>℥</i> |
| TScy;              | U+00426      | Ӑ        |
| tscy;              | U+00446      | ӑ        |
| TSHcy;             | U+0040B      | Ӓ        |
| tshcy;             | U+0045B      | ӓ        |
| Tstrok;            | U+00166      | Ʀ        |
| tstrok;            | U+00167      | »        |
| twixt;             | U+0226C      | ℄        |
| twoheadleftarrow;  | U+0219E      | ↵        |
| twoheadrightarrow; | U+021A0      | ➞        |
| Uacute;            | U+000DA      | Ú        |
| Uacute             | U+000DA      | Ú        |
| uacute;            | U+000FA      | ú        |
| uacute             | U+000FA      | ú        |
| Uarr;              | U+0219F      | ↑        |
| uArr;              | U+021D1      | ↗        |
| uarr;              | U+02191      | ↑        |

| Nom         | Personnages) | Glyphe |
|-------------|--------------|--------|
| Uarrocir;   | U+02949      | ⌘      |
| Ubrcy;      | U+0040E      | Ÿ      |
| ubrcy;      | U+0045E      | ÿ      |
| Ubreve;     | U+0016C      | Ů      |
| ubreve;     | U+0016D      | ů      |
| Ucirc;      | U+000DB      | Ô      |
| Ucirc       | U+000DB      | Ô      |
| ucirc;      | U+000FB      | û      |
| ucirc       | U+000FB      | û      |
| Ucy;        | U+00423      | Ÿ      |
| ucy;        | U+00443      | vous   |
| udarr;      | U+021C5      | ↕      |
| Udblac;     | U+00170      | Ů      |
| udblac;     | U+00171      | ů      |
| udhar;      | U+0296E      | ℔      |
| ufisht;     | U+0297E      | ƚ      |
| Ufr;        | U+1D518      | 𝔲      |
| ufr;        | U+1D532      | 𝔲      |
| Ugrave;     | U+000D9      | Ù      |
| Ugrave      | U+000D9      | Ù      |
| ugrave;     | U+000F9      | ù      |
| ugrave      | U+000F9      | ù      |
| uHar;       | U+02963      | ℔      |
| uharl;      | U+021BF      | ↑      |
| uharr;      | U+021BE      | ↑      |
| uhblk;      | U+02580      | ■      |
| ulcorn;     | U+0231C      | ⌌      |
| ulcorner;   | U+0231C      | ⌌      |
| ulcrop;     | U+0230F      | ⌌      |
| ultri;      | U+025F8      | ⌌      |
| Umacr;      | U+0016A      | ®      |
| umacr;      | U+0016B      | ũ      |
| uml;        | U+000A8      | ¨      |
| uml         | U+000A8      | ¨      |
| UnderBar;   | U+0005F      | —      |
| UnderBrace; | U+023DF      | ⏟      |

| Nom               | Personnages) | Glyphe |
|-------------------|--------------|--------|
| UnderBracket;     | U+023B5      | ⏟      |
| UnderParenthesis; | U+023DD      | ⏟      |
| Union;            | U+022C3      | ∪      |
| UnionPlus;        | U+0228E      | ⋈      |
| Uogon;            | U+00172      | Ů      |
| uogon;            | U+00173      | ø      |
| Uopf;             | U+1D54C      | ℰ      |
| uopf;             | U+1D566      | ℷ      |
| UpArrow;          | U+02191      | ↑      |
| Uparrow;          | U+021D1      | ↗      |
| uparrow;          | U+02191      | ↑      |
| UpArrowBar;       | U+02912      | ↗      |
| UpArrowDownArrow; | U+021C5      | ↕      |
| UpDownArrow;      | U+02195      | ↕      |
| Updownarrow;      | U+021D5      | ↕      |
| updownarrow;      | U+02195      | ↕      |
| UpEquilibrium;    | U+0296E      | ⌚      |
| upharpoonleft;    | U+021BF      | ↵      |
| upharpoonright;   | U+021BE      | ↶      |
| uplus;            | U+0228E      | ⋈      |
| UpperLeftArrow;   | U+02196      | ↖      |
| UpperRightArrow;  | U+02197      | ↗      |
| Upsi;             | U+003D2      | Υ      |
| upsi;             | U+003C5      | υ      |
| upsih;            | U+003D2      | Υ      |
| Upsilon;          | U+003A5      | Υ      |
| upsilon;          | U+003C5      | υ      |
| UpTee;            | U+022A5      | ⊥      |
| UpTeeArrow;       | U+021A5      | ↗      |
| upuparrows;       | U+021C8      | ↗      |
| urcorn;           | U+0231D      | ⌞      |
| urcorner;         | U+0231D      | ⌞      |
| urcrop;           | U+0230E      | ⌞      |
| Uring;            | U+0016E      | Ů      |
| uring;            | U+0016F      | ù      |
| urtri;            | U+025F9      | ⌞      |

| Nom               | Personnages)    | Glyphe |
|-------------------|-----------------|--------|
| Uscr;             | U+1D4B0         | ℳ      |
| uscr;             | U+1D4CA         | ℹ      |
| utdot;            | U+022F0         | ∴      |
| Utilde;           | U+00168         | ×      |
| utilde;           | U+00169         | ÷      |
| utri;             | U+025B5         | △      |
| utrif;            | U+025B4         | ▲      |
| uuarr;            | U+021C8         | ↕      |
| Uuml;             | U+000DC         | Ü      |
| Uuml              | U+000DC         | Ü      |
| uuml;             | U+000FC         | ü      |
| uuml              | U+000FC         | ü      |
| uwangle;          | U+029A7         | ⋈      |
| vangrt;           | U+0299C         | ℳ      |
| varepsilon;       | U+003F5         | ε      |
| varkappa;         | U+003F0         | κ      |
| varnothing;       | U+02205         | ∅      |
| varphi;           | U+003D5         | φ      |
| varpi;            | U+003D6         | π      |
| varpropto;        | U+0221D         | ∝      |
| vArr;             | U+021D5         | ↕      |
| varr;             | U+02195         | ↕      |
| varrho;           | U+003F1         | ϱ      |
| varsigma;         | U+003C2         | ς      |
| varsubsetneq;     | U+0228A U+0FE00 | ⊄      |
| varsubsetneqq;    | U+02ACB U+0FE00 | ⊈      |
| varsupsetneq;     | U+0228B U+0FE00 | ⊅      |
| varsupsetneqq;    | U+02ACC U+0FE00 | ⊉      |
| vartheta;         | U+003D1         | θ      |
| vartriangleleft;  | U+022B2         | ◁      |
| vartriangleright; | U+022B3         | ▷      |
| Vbar;             | U+02AEB         | ℳ      |
| vBar;             | U+02AE8         | ±      |
| vBarv;            | U+02AE9         | ±      |
| Vcy;              | U+00412         | В      |
| vcy;              | U+00432         | в      |

| Nom                | Personnages)    | Glyphe |
|--------------------|-----------------|--------|
| VDash;             | U+022AB         | ⋈      |
| Vdash;             | U+022A9         | ⋇      |
| vDash;             | U+022A8         | ⋆      |
| vdash;             | U+022A2         | ⋔      |
| Vdashl;            | U+02AE6         | ⋈̸     |
| Vee;               | U+022C1         | ∇      |
| vee;               | U+02228         | ∨      |
| veebar;            | U+022BB         | ⋈̸     |
| veeeq;             | U+0225A         | ≍      |
| vellip;            | U+022EE         | ⋮      |
| Verbar;            | U+02016         | ‖      |
| verbar;            | U+0007C         |        |
| Vert;              | U+02016         | ‖      |
| vert;              | U+0007C         |        |
| VerticalBar;       | U+02223         |        |
| VerticalLine;      | U+0007C         |        |
| VerticalSeparator; | U+02758         |        |
| VerticalTilde;     | U+02240         | ⋈̸     |
| VeryThinSpace;     | U+0200A         |        |
| Vfr;               | U+1D519         | Ⅎ      |
| vfr;               | U+1D533         | ℳ      |
| vltri;             | U+022B2         | ⋈̸     |
| vnsup;             | U+02282 U+020D2 | ⊄     |
| vnsup;             | U+02283 U+020D2 | ⊅     |
| Vopf;              | U+1D54D         | Ⅎ      |
| vopf;              | U+1D567         | ℳ      |
| vprop;             | U+0221D         | ⋈̸     |
| vrtri;             | U+022B3         | ⋈̸     |
| Vscr;              | U+1D4B1         | Ⅎ      |
| vscr;              | U+1D4CB         | ℳ      |
| vsubnE;            | U+02ACB U+0FE00 | ⋈̸     |
| vsubne;            | U+0228A U+0FE00 | ⋈̸     |
| vsupnE;            | U+02ACC U+0FE00 | ⋈̸     |
| vsupne;            | U+0228B U+0FE00 | ⋈̸     |
| Vvdash;            | U+022AA         | ⋈̸     |
| vzigzag;           | U+0299A         | ⋈̸     |

| Nom     | Personnages) | Glyphe |
|---------|--------------|--------|
| Wcirc;  | U+00174      | Ŵ      |
| wcirc;  | U+00175      | ŵ      |
| wedbar; | U+02A5F      | ⏟      |
| Wedge;  | U+022C0      | ⋈      |
| wedge;  | U+02227      | ⋀      |
| wedgeq; | U+02259      | ⋴      |
| weierp; | U+02118      | ℘      |
| Wfr;    | U+1D51A      | Ⅎ      |
| wfr;    | U+1D534      | ℳ      |
| Wopf;   | U+1D54E      | ℴ      |
| wopf;   | U+1D568      | ℵ      |
| wp;     | U+02118      | ℘      |
| wr;     | U+02240      | ℳ      |
| wreath; | U+02240      | ℳ      |
| Wscr;   | U+1D4B2      | ℳ      |
| wscr;   | U+1D4CC      | ℴ      |
| xcap;   | U+022C2      | ⋈      |
| xcirc;  | U+025EF      | ⊙      |
| xcup;   | U+022C3      | ⋈      |
| xdtri;  | U+025BD      | ∇      |
| Xfr;    | U+1D51B      | Ⅎ      |
| xfr;    | U+1D535      | ℳ      |
| xhArr;  | U+027FA      | ↔      |
| xharr;  | U+027F7      | ↔      |
| Xi;     | U+0039E      | Ξ      |
| xi;     | U+003BE      | ξ      |
| xlArr;  | U+027F8      | ⇐      |
| xlarr;  | U+027F5      | ⇐      |
| xmap;   | U+027FC      | ↦      |
| xnis;   | U+022FB      | ∋      |
| xodot;  | U+02A00      | ⊙      |
| Xopf;   | U+1D54F      | ℴ      |
| xopf;   | U+1D569      | ℵ      |
| xoplus; | U+02A01      | ⊕      |
| xotime; | U+02A02      | ⊗      |
| xrArr;  | U+027F9      | ⇒      |

| Nom     | Personnages) | Glyphe |
|---------|--------------|--------|
| xrarr;  | U+027F6      | →      |
| Xscr;   | U+1D4B3      | ℳ      |
| xscr;   | U+1D4CD      | ℵ      |
| xsqcup; | U+02A06      | ⊔      |
| xuplus; | U+02A04      | ⊕      |
| xutri;  | U+025B3      | △      |
| xvee;   | U+022C1      | ∨      |
| xwedge; | U+022C0      | ∧      |
| Yacute; | U+000DD      | Ÿ      |
| Yacute  | U+000DD      | Ÿ      |
| yacute; | U+000FD      | ý      |
| yacute  | U+000FD      | ý      |
| YAcy;   | U+0042F      | Ɓ      |
| yacy;   | U+0044F      | ƚ      |
| Ycirc;  | U+00176      | Ŷ      |
| ycirc;  | U+00177      | ŷ      |
| Ycy;    | U+0042B      | Ɓ      |
| ycy;    | U+0044B      | ƚ      |
| yen;    | U+000A5      | ¥      |
| yen     | U+000A5      | ¥      |
| Yfr;    | U+1D51C      | Ƴ      |
| yfr;    | U+1D536      | ƴ      |
| YIcy;   | U+00407      | Ɓ      |
| yicy;   | U+00457      | ƚ      |
| Yopf;   | U+1D550      | Ƴ      |
| yopf;   | U+1D56A      | ƴ      |
| Yscr;   | U+1D4B4      | ℴ      |
| yscr;   | U+1D4CE      | ℵ      |
| YUcy;   | U+0042E      | Ɓ      |
| yucy;   | U+0044E      | ƚ      |
| Yuml;   | U+00178      | Ÿ      |
| yuml;   | U+000FF      | ÿ      |
| yuml    | U+000FF      | ÿ      |
| Zacute; | U+00179      | Ź      |
| zacute; | U+0017A      | ź      |
| Zcaron; | U+0017D      | Ž      |



| Nom             | Personnages) | Glyphe |
|-----------------|--------------|--------|
| zcaron;         | U+0017E      | ž      |
| Zcy;            | U+00417      | З      |
| zcy;            | U+00437      | з      |
| Zdot;           | U+0017B      | Ž      |
| zdot;           | U+0017C      | ž      |
| zeetrf;         | U+02128      | З      |
| ZeroWidthSpace; | U+0200B      |        |
| Zeta;           | U+00396      | Z      |
| zeta;           | U+003B6      | ζ      |
| Zfr;            | U+02128      | З      |
| zfr;            | U+1D537      | ƶ      |
| ZHcy;           | U+00416      | Æ      |
| zhcy;           | U+00436      | Ö      |
| zigrarr;        | U+021DD      | ↯      |
| Zopf;           | U+02124      | Z      |
| zopf;           | U+1D56B      | Ƶ      |
| Zscr;           | U+1D4B5      | ℤ      |
| zscr;           | U+1D4CF      | ℤ      |
| zwj;            | U+0200D      |        |
| zwnj;           | U+0200C      |        |

Ces données sont également disponibles [sous forme de fichier JSON](#) .

*Les glyphes affichés ci-dessus sont non normatifs. Reportez-vous à Unicode pour les définitions formelles des caractères répertoriés ci-dessus.*

*Les noms de référence de caractère proviennent des définitions d'entité XML pour les caractères , bien que seul ce qui précède soit considéré comme normatif. [\[XMLENTITÉ\]](#)  
 Cette liste est statique et [ne sera ni élargie ni modifiée à l'avenir](#) .*

## 1. [14 La syntaxe XML](#)

1. [14.1 Écrire des documents dans la syntaxe XML](#)
2. [14.2 Analyser des documents XML](#)
3. [14.3 Sérialisation des fragments XML](#)
4. [14.4 Analyser des fragments XML](#)

## 14 La syntaxe XML



*Cette section décrit uniquement les règles pour les ressources XML. Les règles pour [text/html](#) les ressources sont discutées dans la section ci-dessus intitulée « [La syntaxe HTML](#) ».*

### 14.1 Écrire des documents dans la syntaxe XML

*La syntaxe XML pour HTML était auparavant appelée "XHTML", mais cette spécification n'utilise pas ce terme (entre autres raisons, car aucun terme de ce type n'est utilisé pour les syntaxes HTML de MathML et SVG).*

La syntaxe pour XML est définie dans *XML* et *les espaces de noms dans XML*. [\[XML\]](#) [\[XMLNS\]](#)

Cette spécification ne définit aucune exigence au niveau de la syntaxe au-delà de celles définies pour XML proprement dit.

Les documents XML peuvent contenir un `DOCTYPE` si vous le souhaitez, mais cela n'est pas obligatoire pour se conformer à cette spécification. Cette spécification ne définit pas d'identifiant public ou système, ni ne fournit de DTD formelle.

*Selon XML, il n'est pas garanti que les processeurs XML traitent le sous-ensemble DTD externe référencé dans le DOCTYPE. Cela signifie, par exemple, que l'utilisation [de références d'entité](#) pour les caractères dans les documents XML n'est pas sûre si elles sont définies dans un fichier externe (sauf pour `&lt;`, `&gt;`, `&amp;`, `&quot;` et `&apos;`).*

### 14.2 Analyser des documents XML

Cette section décrit la relation entre XML et le DOM, avec un accent particulier sur la façon dont cela interagit avec HTML.

Un **analyseur XML**, pour les besoins de cette spécification, est une construction qui suit les règles données en *XML* pour mapper une chaîne d'octets ou de caractères dans un [Document](#) objet.

*Au moment d'écrire ces lignes, aucune règle de ce type n'existe réellement.*

Un [analyseur XML](#) est soit associé à un [Document](#) objet lors de sa création, soit en crée un implicitement.

Celui-ci [Document](#) doit ensuite être rempli avec des nœuds DOM qui représentent la structure arborescente de l'entrée transmise à l'analyseur, telle que définie par XML , *Namespaces in XML* et *DOM* . Lors de la création de nœuds DOM représentant des éléments, la [création d'un élément pour un algorithme de jeton ou un équivalent qui fonctionne sur les structures de données XML appropriées doit être utilisée, pour s'assurer que les interfaces d'élément](#) appropriées sont créées et que [les éléments personnalisés](#) sont configurés correctement.

Les événements de mutation DOM ne doivent pas se déclencher pour les opérations que l' [analyseur XML](#) effectue sur l' [Document](#) arborescence de , mais l'agent utilisateur doit agir comme si les éléments et les attributs étaient ajoutés et définis individuellement afin de déclencher les règles de cette spécification concernant ce qui se passe lorsqu'un L'élément est inséré dans un document ou a ses attributs définis, et les exigences de *DOM concernant* [les observateurs de mutation](#) signifient que les observateurs de mutation *sont* déclenchés (contrairement aux événements de mutation). [\[XML\]](#) [\[XMLNS\]](#) [\[DOM\]](#) [\[UIEVENTS\]](#)

Entre le moment où la balise de début d'un élément est analysée et le moment où la balise de fin de l'élément est analysée ou que l'analyseur détecte une erreur de bonne formation, l'agent utilisateur doit agir comme si l'élément était dans une pile d'éléments [ouverts](#) .

*Ceci est utilisé par divers éléments pour ne démarrer certains processus qu'une fois qu'ils sont sortis de la [pile d'éléments ouverts](#) .*

Cette spécification fournit les informations supplémentaires suivantes que les agents utilisateurs doivent utiliser lors de la récupération d'une entité externe : les identifiants publics donnés dans la liste suivante correspondent tous à [l'URL donnée par ce lien](#) . (Cette URL est une DTD contenant les [déclarations d'entité](#) pour les noms répertoriés dans la section [des références de caractères nommés](#) .) [\[XML\]](#)

- `-//W3C//DTD XHTML 1.0 Transitional//EN`
- `-//W3C//DTD XHTML 1.1//EN`
- `-//W3C//DTD XHTML 1.0 Strict//EN`
- `-//W3C//DTD XHTML 1.0 Frameset//EN`
- `-//W3C//DTD XHTML Basic 1.0//EN`
- `-//W3C//DTD XHTML 1.1 plus MathML 2.0//EN`
- `-//W3C//DTD XHTML 1.1 plus MathML 2.0 plus SVG 1.1//EN`
- `-//W3C//DTD MathML 2.0//EN`
- `-//WAPFORUM//DTD XHTML Mobile 1.0//EN`

De plus, les agents utilisateurs devraient tenter de récupérer le contenu de l'entité externe ci-dessus lorsque l'un des identifiants publics ci-dessus est utilisé, et ne devraient pas tenter de récupérer le contenu d'une autre entité externe.

*Ce n'est pas strictement une [violation](#) de XML , mais cela contredit l'esprit des exigences de XML . Ceci est motivé par un désir pour les agents utilisateurs de gérer tous les entités d'une manière interopérable sans nécessiter d'accès au réseau pour gérer les sous-ensembles externes. [\[XML\]](#)*

Les analyseurs XML peuvent être appelés avec **la prise en charge des scripts XML activée** ou **la prise en charge des scripts XML désactivée**. Sauf indication contraire, les analyseurs XML sont appelés avec la prise en charge des scripts XML activée.

Lorsqu'un analyseur XML avec la prise en charge des scripts XML activé crée un script élément, il doit avoir son ensemble de documents d'analyseur et sa force asynchrone définie sur false. Si l'analyseur a été créé dans le cadre de l' algorithme d'analyse de fragment XML, l'élément déjà démarré doit être défini sur true. Lorsque la balise de fin de l'élément est ensuite analysée, l'agent utilisateur doit effectuer un point de contrôle de microtâche, puis préparer l' script élément. Si cela entraîne la présence d'un script de blocage d'analyse en attente, l'agent utilisateur doit exécuter les étapes suivantes :

1. Bloquez cette instance de l' analyseur XML, de sorte que la boucle d'événements n'exécute pas les tâches qui l'invoquent.
2. Faites tourner la boucle d'événements jusqu'à ce que l'analyseur Document n'ait plus de feuille de style bloquant les scripts et que le script de blocage d'analyse en attente prêt à être exécuté par l'analyseur soit vrai.
3. Débloquez cette instance de l'analyseur XML, de sorte que les tâches qui l'invoquent puissent à nouveau être exécutées.
4. Exécutez l'élément de script fourni par le script de blocage d'analyse en attente.
5. Définissez le script de blocage d'analyse en attente sur null.

*Puisque l' `document.write()` API n'est pas disponible pour les documents XML, une grande partie de la complexité de l' analyseur HTML n'est pas nécessaire dans l' analyseur XML.*

*Lorsque l' analyseur XML a désactivé la prise en charge des scripts XML, rien de tout cela ne se produit.*

Lorsqu'un analyseur XML ajouterait un nœud à un template élément, il doit à la place l'ajouter au contenu du modèle template de l'élément (un nœud). DocumentFragment

*Il s'agit d'une violation délibérée de XML ; Malheureusement, XML n'est pas formellement extensible de la manière nécessaire pour template le traitement. [XML]*

Lorsqu'un analyseur XML crée un Node objet, son document de nœud doit être défini sur le document de nœud du nœud dans lequel le nœud nouvellement créé doit être inséré.

Certains algorithmes de cette spécification **alimentent à la cuillère les** caractères de l'analyseur une chaîne à la fois. Dans de tels cas, l' analyseur XML doit agir comme il l'aurait fait s'il était confronté à une seule chaîne constituée de la concaténation de tous ces caractères.

Lorsqu'un [analyseur XML](#) atteint la fin de son entrée, il doit [arrêter l'analyse](#) , en suivant les mêmes règles que l' [analyseur HTML](#) . Un [parseur XML](#) peut également être [abandonné](#) , ce qui doit à nouveau être fait de la même manière que pour un [parseur HTML](#) .

Pour les besoins des vérificateurs de conformité, si une ressource est déterminée comme étant dans [la syntaxe XML](#) , il s'agit alors d'un [document XML](#) .

### 14.3 Sérialisation des fragments XML

L' **algorithme de sérialisation de fragment XML** pour un [Document](#) nœud [Element](#) ou renvoie un fragment de XML qui représente ce nœud ou lève une exception.

Pour [Documents](#), l'algorithme doit renvoyer une chaîne sous la forme d'une [entité de document](#) , si aucun des cas d'erreur ci-dessous ne s'applique.

Pour [Elements](#), l'algorithme doit renvoyer une chaîne sous la forme d'une [entité analysée générale interne](#) , si aucun des cas d'erreur ci-dessous ne s'applique.

Dans les deux cas, la chaîne renvoyée doit être bien formée dans l'espace de noms XML et doit être une sérialisation isomorphe de tous les [nœuds enfants pertinents](#) de ce nœud , dans [l'ordre de l'arborescence](#) . Les agents utilisateurs peuvent ajuster les préfixes et les déclarations d'espace de noms dans la sérialisation (et peuvent en effet être forcés de le faire dans certains cas pour obtenir un XML bien formé d'espace de noms). Les agents utilisateurs peuvent utiliser une combinaison de texte normal et de références de caractères pour représenter [Text](#) des nœuds dans le DOM.

**Les nœuds enfants pertinents** d'un nœud sont ceux qui s'appliquent compte tenu des règles suivantes :

#### Pour [template](#) les éléments

Les [nœuds enfants pertinents](#) sont les nœuds enfants du [contenu du modèle](#) [template](#) de l'élément , le cas échéant.

#### Pour tous les autres nœuds

Les [nœuds enfants pertinents](#) sont les nœuds enfants du nœud lui-même, le cas échéant.

Pour [Elements](#), si l'un des éléments de la sérialisation ne se trouve dans aucun espace de noms, l'espace de noms par défaut dans la portée de ces éléments doit être explicitement déclaré comme chaîne vide. (Cela ne s'applique pas dans le [Document](#) cas.) [\[XML\]](#) [\[XMLNS\]](#)

Pour les besoins de cette section, une entité analysée générale interne est considérée comme bien formée avec un espace de noms XML si un document

composé d'un élément sans déclaration d'espace de noms dont le contenu est l'entité analysée générale interne serait lui-même bien formé avec un espace de noms XML.

Si l'un des cas d'erreur suivants est détecté dans la sous-arborescence DOM en cours de sérialisation, l'algorithme doit lancer

un `"InvalidStateException" DOMException` au lieu de renvoyer une chaîne :

- Un `Document` nœud sans nœuds d'éléments enfants.
- Un `DocumentType` nœud qui a un identificateur public de sous-ensemble externe qui contient des caractères qui ne correspondent pas à la `PubidChar` production XML. [\[XML\]](#)
- Un `DocumentType` nœud qui a un identificateur de système de sous-ensemble externe qui contient à la fois un guillemet U+0022 (") et un APOSTROPHE U+0027 (') ou qui contient des caractères qui ne correspondent pas à la production XML. [\[ CharXML \]](#)
- Un nœud avec un nom local contenant un U+003A COLON (:).
- Un nœud avec un nom local qui ne correspond pas à la `Name` production XML. [\[XML\]](#)
- Un `Attr` nœud sans espace de noms dont le nom local est la chaîne en minuscules " xmlns ". [\[XMLNS\]](#)
- Un `Element` nœud avec deux ou plusieurs attributs avec le même nom local et le même espace de noms.
- Un `Attr` nœud, `Text` un nœud, `Comment` un nœud ou `ProcessingInstruction` un nœud dont les données contiennent des caractères qui ne correspondent pas à la `Char` production XML. [\[XML\]](#)
- Un `Comment` nœud dont les données contiennent deux caractères U+002D HYPHEN-MINUS adjacents (-) ou se terminent par un tel caractère.
- Un `ProcessingInstruction` nœud dont le nom cible est une correspondance [ASCII non sensible à la casse](#) pour la chaîne " xml ".
- Un `ProcessingInstruction` nœud dont le nom cible contient un U+003A COLON (:).
- Un `ProcessingInstruction` nœud dont les données contiennent la chaîne " ?> ".

*Ce sont les seuls moyens de rendre un DOM non sérialisable. Le DOM applique toutes les autres contraintes XML ; par exemple, essayer d'ajouter deux éléments à un `Document` nœud lancera un `"HierarchyRequestError" DOMException` .*

## 14.4 Analyser des fragments XML

L' **algorithme d'analyse de fragment XML** renvoie un [Document](#) ou lève un ["SyntaxError" DOMException](#) . Étant donné une *entrée* de chaîne et un élément de [contexte](#) context , l'algorithme est le suivant :

1. Créez un nouvel [analyseur XML](#) .
2. [Alimentez l'analyseur](#) qui vient de créer la chaîne correspondant à la balise de début de l' élément [de contexte](#) , en déclarant tous les préfixes d'espace de noms qui sont dans la portée de cet élément dans le DOM, ainsi qu'en déclarant l'espace de noms par défaut (le cas échéant) qui est dans la portée de cet élément. élément dans le DOM.

Un préfixe d'espace de noms est dans la portée si la `lookupNamespaceURI()` méthode DOM sur l'élément renvoie une valeur non nulle pour ce préfixe.

L'espace de noms par défaut est l'espace de noms pour lequel la `isDefaultNamespace()` méthode DOM sur l'élément renverrait true.

*Non DOCTYPE est passé à l'analyseur, et donc aucun sous-ensemble externe n'est référencé, et donc aucune entité ne sera reconnue.*

3. [Alimentez l'analyseur](#) vient de créer la chaîne *input* .
4. [Alimentez l'analyseur](#) qui vient de créer la chaîne correspondant à la balise de fin de l' élément [de contexte](#) .
5. S'il y a une erreur de bonne formation XML ou d'espace de noms XML, lancez un ["SyntaxError" DOMException](#) .
6. Si l' [élément de document](#) du résultat [Document](#) a des nœuds frères, lancez un ["SyntaxError" DOMException](#) .
7. Renvoie les nœuds enfants de l' [élément document](#) du résultat [Document](#), dans [l'ordre de l'arborescence](#) .

### 1. [15 Rendu](#)

1. [15.1 Présentation](#)
2. [15.2 La feuille de style de l'agent utilisateur CSS et les conseils de présentation](#)
3. [15.3 Éléments non remplacés](#)
  1. [15.3.1 Éléments cachés](#)
  2. [15.3.2 La page](#)
  3. [15.3.3 Contenu du flux](#)



4. [15.3.4 Phrase contenu](#)
5. [15.3.5 Texte bidirectionnel](#)
6. [15.3.6 Sections et titres](#)
7. [15.3.7 Listes](#)
8. [15.3.8 Tableaux](#)
9. [15.3.9 bizarreries d'effondrement de marge](#)
10. [15.3.10 Champs de formulaire](#)
11. [15.3.11 L'`hr`élément](#)
12. [15.3.12 Les éléments `fieldset` et `legend`](#)
4. [15.4 Éléments remplacés](#)
  1. [15.4.1 Contenu intégré](#)
  2. [15.4.2 Images](#)
  3. [15.4.3 Attributs pour le contenu intégré et les images](#)
  4. [15.4.4 Images cliquables](#)
5. [15.5 Gadgets](#)
  1. [15.5.1 Apparence native](#)
  2. [15.5.2 Disposition des boutons](#)
  3. [15.5.3 L'`button`élément](#)
  4. [15.5.4 Les éléments `details` et `summary`](#)
  5. [15.5.5 L'`input`élément en tant que widget de saisie de texte](#)
  6. [15.5.6 L'`input`élément en tant que widgets spécifiques à un domaine](#)
  7. [15.5.7 L'`input`élément comme contrôle de distance](#)
  8. [15.5.8 L'`input`élément comme puits de couleur](#)
  9. [15.5.9 L'`input`élément en tant que widgets de case à cocher et de bouton radio](#)
  10. [15.5.10 L'`input`élément en tant que contrôle de téléchargement de fichier](#)
  11. [15.5.11 L'`input`élément sous forme de bouton](#)
  12. [15.5.12 L'`marquee`élément](#)
  13. [15.5.13 L'`meter`élément](#)
  14. [15.5.14 L'`progress`élément](#)
  15. [15.5.15 L'`select`élément](#)
  16. [15.5.16 L'`textarea`élément](#)
6. [15.6 Cadres et jeux de cadres](#)
7. [15.7 Médias interactifs](#)
  1. [15.7.1 Liens, formulaires et navigation](#)



2. [15.7.2 L'titleattribut](#)
3. [15.7.3 Modification des hôtes](#)
4. [15.7.4 Texte rendu dans les interfaces utilisateur natives](#)
8. [15.8 Supports d'impression](#)
9. [15.9 Documents XML sans style](#)

## 15 Rendu

*Les agents utilisateurs ne sont pas tenus de présenter les documents HTML d'une manière particulière. Cependant, cette section fournit un ensemble de suggestions pour le rendu des documents HTML qui, si elles sont suivies, sont susceptibles de conduire à une expérience utilisateur qui ressemble étroitement à l'expérience prévue par les auteurs des documents. Afin d'éviter toute confusion concernant la normativité de cette section, "doit" n'a pas été utilisé. Au lieu de cela, le terme "attendu" est utilisé pour indiquer un comportement qui conduira à cette expérience. Aux fins de conformité pour les agents utilisateurs désignés comme [prenant en charge le rendu par défaut suggéré](#), le terme "attendu" dans cette section a les mêmes implications de conformité que "doit".*

### 15.1 Présentation

Les suggestions de cette section sont généralement exprimées en termes CSS. Les agents utilisateurs sont censés soit prendre en charge CSS, soit traduire les règles CSS données dans cette section en approximations pour d'autres mécanismes de présentation.

En l'absence de règles de couche de style contraires (par exemple, les feuilles de style d'auteur), les agents utilisateurs sont censés restituer un élément de manière à ce qu'il transmette à l'utilisateur la signification que l'élément représente, [comme](#) décrit par la présente spécification.

Les suggestions de cette section supposent généralement un support de sortie visuel avec une résolution de 96 dpi ou plus, mais HTML est destiné à s'appliquer à plusieurs supports (il s'agit d'un langage *indépendant du support*). Les implémenteurs d'agents utilisateurs sont encouragés à adapter les suggestions de cette section à leurs médias cibles.

---

Un élément est **rendu** s'il a des boîtes de mise en page CSS associées, des boîtes de mise en page SVG ou un équivalent dans d'autres langages de style.

*Le simple fait d'être hors écran ne signifie pas que l'élément n'est pas [rendu](#) . La présence de l' [hidden](#) attribut signifie normalement que l'élément n'est pas [rendu](#) , bien que cela puisse être remplacé par les feuilles de style. L' état [pleinement actif](#) n'affecte pas le [rendu](#) d'un élément ou non. Même si un document n'est pas [entièrement actif](#) et n'est pas affiché du tout à l'utilisateur, les éléments qu'il contient peuvent toujours être qualifiés de "en cours de rendu".*

On dit qu'un élément intersecte la **fenêtre** lorsqu'il est [rendu](#) et que sa boîte de mise en page CSS associée intersecte la [fenêtre](#) .

*Semblable à l' état [en cours de rendu](#) , les éléments des documents qui ne sont pas [entièrement actifs](#) peuvent toujours [croiser la fenêtre](#) . La [fenêtre d'affichage](#) n'est pas partagée entre les documents et peut ne pas toujours être affichée à l'utilisateur, de sorte qu'un élément d'un document non [entièrement actif](#) peut toujours croiser la [fenêtre d'affichage](#) associée à son document. Cette spécification ne définit pas le moment précis du test de l'intersection, mais il est suggéré que le moment corresponde à celui de l'API Intersection Observer. [\[OBSERVATEUR D'INTERSECTION\]](#)*

---

Les agents utilisateurs qui ne respectent pas les feuilles de style CSS au niveau de l'auteur doivent néanmoins agir comme s'ils appliquaient les règles CSS données dans ces sections d'une manière cohérente avec cette spécification et les spécifications CSS et Unicode pertinentes. [\[CSS\]](#) [\[UNICODE\]](#) [\[BIDI\]](#)

*Ceci est particulièrement important pour les problèmes liés aux propriétés ['display'](#) , ['unicode-bidi'](#) et ['direction'](#) .*

## 15.2 La feuille de style de l'agent utilisateur CSS et les conseils de présentation

Les règles CSS données dans ces sous-sections sont, sauf indication contraire, censées être utilisées dans le cadre des feuilles de style par défaut au niveau de l'agent utilisateur pour tous les documents contenant des éléments [HTML](#) .

Certaines règles sont destinées à la partie des conseils de présentation à spécificité zéro au niveau de l'auteur de la cascade CSS ; ceux-ci sont explicitement appelés comme **conseils de présentation** .

---

Lorsque le texte ci-dessous indique qu'un *attribut* d'attribut sur un élément *élément* **correspond à la propriété de longueur de pixel** (ou aux propriétés) *properties* , cela signifie que si *l'élément* a un ensemble *d'attributs* d'attribut , et l'analyse de la valeur de cet attribut en utilisant les [règles d'analyse des entiers non négatifs](#) ne génère pas d'erreur, l'agent utilisateur est censé utiliser la valeur analysée comme longueur de pixel pour un [indice de présentation](#) pour *properties* .

Lorsque le texte ci-dessous indique qu'un attribut d'attribut *sur* un élément *élément* **correspond à la propriété de dimension** (ou aux propriétés) *properties* , cela signifie que si *l'élément* a un ensemble *d'attributs* d'attribut et que l'analyse de la valeur de cet attribut à l'aide des [règles d'analyse des valeurs de dimension](#) ne fonctionne pas générer une erreur, alors l'agent utilisateur est censé utiliser la dimension analysée comme valeur d'un [indice de présentation](#) pour *properties* , avec la valeur donnée en tant que longueur de pixel si la dimension était une longueur, et avec la valeur donnée en pourcentage si la dimension était un pourcentage.

Lorsque le texte ci-dessous indique qu'un attribut d'attribut *sur* un élément *élément* **correspond à la propriété de dimension (ignorant zéro)** (ou propriétés) *properties* , cela signifie que si *l'élément* a un ensemble *d'attributs* d'attribut , et l'analyse de la valeur de cet attribut en utilisant les [règles d'analyse non nulle les valeurs de dimension](#) ne génèrent pas d'erreur, alors l'agent utilisateur est censé utiliser la dimension analysée comme valeur d'un [indice de présentation](#) pour *properties* , avec la valeur donnée en tant que longueur de pixel si la dimension était une longueur, et avec la valeur donnée sous forme de pourcentage si la dimension était un pourcentage.

Lorsque le texte ci-dessous indique qu'une paire d'attributs *w* et *h* sur un élément *element* **correspond à la propriété ratio d'aspect** , cela signifie que si *element* a les deux attributs *w* et *h* , et l'analyse des valeurs de ces attributs en utilisant les [règles d'analyse non- les entiers négatifs](#) ne génèrent pas d'erreur pour l'un ou l'autre, alors l'agent utilisateur est censé utiliser les entiers analysés comme indice de [présentation](#) pour la propriété '[aspect-ratio](#)' du formulaire `.auto w / h`

Lorsque le texte ci-dessous indique qu'une paire d'attributs *w* et *h* sur un élément *element* **correspond à la propriété rapport d'aspect (en utilisant les règles de dimension)** , cela signifie que si *l'élément* a les deux attributs *w* et *h* , et l'analyse des valeurs de ces attributs à l'aide de la [Les règles d'analyse des valeurs de dimension](#) ne génèrent pas d'erreur ou ne renvoient pas de pourcentage pour l'un ou l'autre, alors l'agent utilisateur est censé utiliser les dimensions analysées comme indice de [présentation](#) pour la propriété '[aspect-ratio](#)' du formulaire `.auto w / h`

Lorsqu'un agent utilisateur doit **aligner les descendants** d'un nœud, l'agent utilisateur est censé aligner uniquement les descendants qui ont à la fois leurs propriétés '[margin-inline-start](#)' et '[margin-inline-end](#)' calculées sur une valeur

autre que 'auto ', qui sont surcontraints et qui ont une de ces deux marges avec une [valeur usée](#) forcée à une valeur supérieure, et qui n'ont pas eux-mêmes d' `align` attribut applicable. Lorsque plusieurs éléments doivent [aligner](#) un descendant particulier, l'élément le plus profondément imbriqué devrait remplacer les autres. Les éléments alignés sont censés être alignés en ayant les [valeurs utilisées](#) de leurs marges sur la [ligne gauche](#) et les côtés [droits de la ligne](#) soient définis en conséquence. [\[CSSLOGIQUE\]](#) [\[CSSWM\]](#)

## 15.3 Éléments non remplacés

### 15.3.1 Éléments cachés

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
area, base, basefont, datalist, head, link, meta, noembed,
```

```
noframes, param, rp, script, style, template, title {
```

```
display: none;
```

```
}
```

```
[hidden]:not([hidden=until-found i]) {
```

```
display: none;
```

```
}
```

```
[hidden=until-found i]:not(embed) {
```

```
content-visibility: hidden;
```

```
}
```

```
embed[hidden] { display: inline; height: 0; width: 0; }
```

```
input[type=hidden i] { display: none !important; }
```

```
@media (scripting) {
```

```
noscript { display: none !important; }
```

```
}
```

### 15.3.2 La page

```
@namespace url (http://www.w3.org/1999/xhtml);
```

```
html, body { display: block; }
```

Pour chaque propriété du tableau ci-dessous, pour un bodyélément donné, le premier attribut qui existe [correspond à la propriété de longueur en pixels](#) de l' bodyélément. Si aucun des attributs d'une propriété n'est trouvé, ou si la valeur de l'attribut qui a été trouvé ne peut pas être analysée avec succès, une valeur par défaut de 8px devrait être utilisée pour cette propriété à la place.

| Propriété                            | Source                                                                                                |
|--------------------------------------|-------------------------------------------------------------------------------------------------------|
| ' <a href="#">marge supérieure</a> ' | L' attribut <u>body</u> de l'élément <u>marginheight</u>                                              |
|                                      | L' attribut <u>body</u> de l'élément <u>topmargin</u>                                                 |
|                                      | L' attribut de <a href="#">l'élément frame conteneur</a> <u>body</u> de l'élément <u>marginheight</u> |
| ' <a href="#">marge droite</a> '     | L' attribut <u>body</u> de l'élément <u>marginwidth</u>                                               |
|                                      | L' attribut <u>body</u> de l'élément <u>rightmargin</u>                                               |
|                                      | L' attribut de <a href="#">l'élément frame conteneur</a> <u>body</u> de l'élément <u>marginwidth</u>  |
| ' <a href="#">marge inférieure</a> ' | L' attribut <u>body</u> de l'élément <u>marginheight</u>                                              |
|                                      | L' attribut <u>body</u> de l'élément <u>bottommargin</u>                                              |
|                                      | L' attribut de <a href="#">l'élément frame conteneur</a> <u>body</u> de l'élément <u>marginheight</u> |
| ' <a href="#">marge gauche</a> '     | L' attribut <u>body</u> de l'élément <u>marginwidth</u>                                               |
|                                      | L' attribut <u>body</u> de l'élément <u>leftmargin</u>                                                |
|                                      | L' attribut de <a href="#">l'élément frame conteneur</a> <u>body</u> de l'élément <u>marginwidth</u>  |

Si le [nœud navigable](#) du [nœud document](#)`body` de l'élément est un [navigable enfant](#) et que le [conteneur](#) de ce [navigable](#) est un élément or , alors l' **élément frame** **conteneur** de l' élément est cet élément or . Sinon, il n'y a pas [d'élément de cadre de conteneur](#) .`frameiframebodyframeiframe`

**Les exigences ci-dessus impliquent qu'une page peut modifier les marges d'une autre page (y compris une d'une autre [origine](#) ) en utilisant, par exemple, un [iframe](#). Il s'agit potentiellement d'un risque de sécurité, car cela pourrait dans certains cas permettre à une attaque de créer une situation dans laquelle une page n'est pas rendue comme l'auteur l'avait prévu, éventuellement à des fins de phishing ou d'induire l'utilisateur en erreur.**

---

Si le [nœud navigable](#)`Document` d'un est un [enfant navigable](#) , alors on s'attend à ce qu'il soit positionné et dimensionné pour tenir à l'intérieur de la [boîte de contenu](#) du [conteneur](#) de ce [navigable](#) . Si le [conteneur](#) n'est pas [rendu](#) , le [navigable](#) devrait avoir une [fenêtre d'affichage](#) avec une largeur et une hauteur nulles.

Si le [nœud navigable](#)`Document` d' un est un [enfant navigable](#) , le [conteneur](#) de cet élément [navigable](#) est un élément ou , cet élément a un attribut et la valeur de cet attribut est une correspondance [ASCII insensible à la casse](#) pour la chaîne " ", " " ou " ", alors l'agent utilisateur est censé empêcher l'affichage de toute barre de défilement pour la [fenêtre d'affichage](#) du nœud [navigable](#) , quelle que soit la propriété ['overflow'](#) qui s'applique à cette [fenêtre](#) .`frameiframescrollingoffnoscrollnoDocument`

---

Lorsqu'un [body](#)élément a un [background](#) attribut défini sur une valeur non vide, la nouvelle valeur est censée être [analysée](#) par rapport au [nœud document](#) de l'élément , et si cela réussit, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation](#) définissant le propriété ['background-image'](#) de l'élément à la [chaîne d'URL résultante](#) .

Lorsqu'un [body](#)élément a un [bgcolor](#) ensemble d'attributs, la nouvelle valeur doit être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#) , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation](#) définissant le propriété ['background-color'](#) de l'élément à la couleur résultante.

Lorsqu'un `body` élément a un `text` attribut, sa valeur est censée être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#) , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation définissant le ' de l'élément `color` à la couleur résultante.](#)

Lorsqu'un `body` élément a un `link` attribut, sa valeur doit être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#) , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation](#) définissant la `'couleur'` propriété de tout élément de la `Document` correspondance de la `:link` [pseudo-classe](#) à la couleur résultante.

Lorsqu'un `body` élément a un `vlink` attribut, sa valeur doit être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#) , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation](#) définissant la `'couleur'` propriété de tout élément de la `Document` correspondance de la `:visited` [pseudo-classe](#) à la couleur résultante.

Lorsqu'un `body` élément a un `alink` attribut, sa valeur doit être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#) , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation](#) définissant la `'couleur'` propriété de tout élément dans la `Document` correspondance de la `:active` [pseudo-classe](#) et de la `:link` [pseudo-classe](#) ou de la `:visited` [pseudo-classe](#) avec la couleur résultante.

### 15.3.3 Contenu du flux

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
address, blockquote, center, dialog, div, figure, figcaption,
```

```
footer, form,
```

```
header, hr, legend, listing, main, p, plaintext, pre, xmp {
```

```
display: block;
```

```
}
```

```
blockquote, figure, listing, p, plaintext, pre, xmp {
```

```
margin-block-start: 1em; margin-block-end: 1em;
```

```
}
```

```
blockquote, figure { margin-inline-start: 40px; margin-inline-  
end: 40px; }
```

```
address { font-style: italic; }
```

```
listing, plaintext, pre, xmp {
```

```
font-family: monospace; white-space: pre;
```

```
}
```

```
dialog:not([open]) { display: none; }
```

```
dialog {
```

```
position: absolute;
```

```
inset-inline-start: 0; inset-inline-end: 0;
```

```
width: fit-content;
```

```
height: fit-content;
```

```
margin: auto;
```

```
border: solid;
```

```
padding: 1em;
```

```
background-color: Canvas;
```

```
color: CanvasText;
```

```
}
```

```
dialog::backdrop {
```

```
background: rgba(0,0,0,0.1);
```



```
}
```

```
[popover]:closed:not(dialog[open]) {
```

```
display:none;
```

```
}
```

```
dialog[popover]:not(:closed) {
```

```
display:block;
```

```
}
```

```
[popover] {
```

```
position: fixed;
```

```
inset: 0;
```

```
width: fit-content;
```

```
height: fit-content;
```

```
margin: auto;
```

```
border: solid;
```

```
padding: 0.25em;
```

```
overflow: auto;
```

```
color: CanvasText;
```

```
background-color: Canvas;
```

```
}
```

```
[popover]:open::backdrop {
```

```
position: fixed;
```

```
inset: 0;
```

```
pointer-events: none !important;
```

```
background-color: transparent;
```

```
}
```

```
slot {
```

```
display: contents;
```

```
}
```

Les règles suivantes sont également censées s'appliquer, comme [conseils de présentation](#) :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
pre[wrap] { white-space: pre-wrap; }
```

En [mode quirks](#) , les règles suivantes sont également censées s'appliquer :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
form { margin-block-end: 1em; }
```

---

L' [center](#) élément, et l' [div](#) élément lorsqu'il a un [align](#) attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " [center](#) ou la chaîne " [middle](#)", sont censés centrer le texte sur eux-mêmes, comme s'ils avaient leur '[text-align](#)' propriété définie sur 'center' dans un [conseil de présentation](#) et pour [aligner les descendants](#) sur le centre.

L' [div](#) élément, lorsqu'il a un [align](#) attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " [left](#)", est censé aligner

le texte à gauche sur lui-même, comme si sa propriété `'text-align'` était définie sur 'left' dans un [indice de présentation](#) et pour [aligner les descendants](#) à gauche.

L' `div`élément, lorsqu'il a un `align` attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " `right`", est censé aligner le texte à droite sur lui-même, comme si sa propriété `'text-align'` était définie sur 'right' dans un [indice de présentation](#) et pour [aligner les descendants](#) à droite.

L' `div`élément, lorsqu'il a un `align` attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " `justify`", est censé justifier complètement le texte en lui-même, comme si sa propriété `'text-align'` était définie sur 'justify' dans un [indice de présentation](#) et pour [aligner les descendants](#) à gauche.

L' `dialog`élément, lorsque son indicateur [est modal](#) est vrai, doit agir comme s'il avait une règle de feuille de style au niveau de l'agent utilisateur définissant les propriétés suivantes :

- [propriété 'position'](#) à 'fixe'
- [propriété 'overflow'](#) à 'auto'
- [propriété 'inset-block-start'](#) à '0'
- [propriété 'inset-block-end'](#) à '0'
- [propriété 'max-width'](#) à 'calc(100% - 6px - 2em)'
- [propriété 'max-height'](#) à 'calc(100% - 6px - 2em)'

#### 15.3.4 Phrase contenu

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
cite, dfn, em, i, var { font-style: italic; }
```

```
b, strong { font-weight: bolder; }
```

```
code, kbd, samp, tt { font-family: monospace; }
```

```
big { font-size: larger; }
```

```
small { font-size: smaller; }
```

```
sub { vertical-align: sub; }
```

```
sup { vertical-align: super; }
```

```
sub, sup { line-height: normal; font-size: smaller; }
```

```
ruby { display: ruby; }
```

```
rt { display: ruby-text; }
```

```
:link { color: #0000EE; }
```

```
:visited { color: #551A8B; }
```

```
:link:active, :visited:active { color: #FF0000; }
```

```
:link, :visited { text-decoration: underline; cursor: pointer; }
```

```
:focus-visible { outline: auto; }
```

```
mark { background: yellow; color: black; } /* this color is just
```

```
a suggestion and can be changed based on implementation feedback
```

```
*/
```

```
abbr[title], acronym[title] { text-decoration: dotted underline;
```

```
}
```

```
ins, u { text-decoration: underline; }
```

```
del, s, strike { text-decoration: line-through; }
```

```
q::before { content: open-quote; }
```

```
q::after { content: close-quote; }
```

```
br { display-outside: newline; } /* this also has bidi
```

```
implications */
```

```
nobr { white-space: nowrap; }
```

```
wbr { display-outside: break-opportunity; } /* this also has bidi
```

```
implications */
```

```
nobr wbr { white-space: normal; }
```

Les règles suivantes sont également censées s'appliquer, comme [conseils de présentation](#) :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
br[clear=left i] { clear: left; }
```

```
br[clear=right i] { clear: right; }
```

```
br[clear=all i], br[clear=both i] { clear: both; }
```

Pour les besoins du modèle CSS ruby, les séries d'enfants d'[ruby](#) éléments qui ne le sont pas [rt](#) ou [rp](#) les éléments doivent être enveloppés dans des boîtes anonymes dont la propriété '[display](#)' a la valeur '[ruby-base](#)'. [\[CSSRUBY\]](#)

Lorsqu'une partie particulière d'un ruby a plus d'une annotation, les annotations doivent être réparties des deux côtés du texte de base afin de minimiser l'empilement des annotations ruby sur un côté.

*Lorsqu'il sera possible de le faire, l'exigence précédente sera mise à jour pour être exprimée en termes de CSS ruby. (Actuellement, CSS ruby ne gère pas [ruby](#) les éléments imbriqués ou plusieurs [rt](#) éléments séquentiels, c'est ainsi que cette sémantique est exprimée.)*

Les agents utilisateurs qui ne prennent pas en charge le rendu ruby correct sont censés afficher des parenthèses autour du texte des [rt](#) éléments en l'absence d'[rp](#) éléments.

---

Les agents utilisateurs sont censés prendre en charge la propriété '[clear](#)' sur les éléments en ligne (afin de rendre [br](#) les éléments avec [clear](#) des attributs) de la manière décrite dans la note non normative à cet effet dans CSS .

La valeur initiale de la propriété '[color](#)' devrait être noire. La valeur initiale de la propriété '[background-color](#)' devrait être 'transparent'. Le fond de la toile devrait être blanc.

---

Lorsqu'un [font](#) élément a un [color](#) attribut, sa valeur est censée être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#) , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation définissant le '](#) de l'élément [color](#)' à la couleur résultante.

Lorsqu'un [font](#) élément a un [face](#) attribut, l'agent utilisateur est censé traiter l'attribut comme une [indication de présentation](#) définissant la propriété '[font-family](#)' de l'élément à la valeur de l'attribut.

Lorsqu'un [font](#) élément a un [size](#) attribut, l'agent utilisateur doit suivre les étapes suivantes, connues sous le nom de **règles d'analyse d'une taille de police héritée** , pour traiter l'attribut comme un [indice de présentation](#) définissant la propriété '[font-size](#)' de l'élément :

1. Soit *input* la valeur de l'attribut.
2. Soit *position* un pointeur dans *input* , pointant initialement au début de la chaîne.
3. [Ignorer les espaces blancs ASCII](#) dans *la position d'entrée* donnée .
4. Si *la position* dépasse la fin de *l'entrée* , il n'y a pas [d'indice de présentation](#) . Retour.
5. Si le caractère à *la position* est un caractère U + 002B PLUS SIGN (+), alors laissez *mode* être *relatif-plus* et avancez *la position* au caractère suivant. Sinon, si le caractère à *la position* est un caractère U + 002D HYPHEN-MINUS (-), alors laissez *mode* être *relatif-moins* et avancez *la position* au caractère suivant. Sinon, laissez *mode* être *absolu* .
6. [Collectez une séquence de points de code](#) qui sont [des chiffres ASCII](#) à partir de *la position* donnée en *entrée* et laissez la séquence résultante être *des chiffres* .
7. Si *chiffres* est la chaîne vide, il n'y a pas [d'indication de présentation](#) . Retour.

8. Interpréter *les chiffres* comme un entier en base dix. Soit *value* le nombre résultant.
9. Si *mode* est *relatif-plus* , alors incrémentez *la valeur* de 3. Si *mode* est *relatif-moins* , alors laissez *la valeur* être le résultat de la soustraction de *la valeur* de 3.
10. Si *la valeur* est supérieure à 7, laissez-le être 7.
11. Si *la valeur* est inférieure à 1, soit 1.
12. Définissez 'font-size' sur le mot-clé correspondant à la valeur de *value* selon le tableau suivant :

| <i>value</i> | mot clé <u>'font-size'</u> |
|--------------|----------------------------|
| 1            | 'x-petit'                  |
| 2            | 'petit'                    |
| 3            | 'moyen'                    |
| 4            | 'grand'                    |
| 5            | 'x-large'                  |
| 6            | 'xx-large'                 |
| 7            | 'xxx-large'                |

### 15.3.5 Texte bidirectionnel

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
[dir]:dir(ltr), bdi:dir(ltr), input[type=tel i]:dir(ltr) {
```

```
direction: ltr; }
```

```
[dir]:dir rtl), bdi:dir rtl) { direction: rtl; }
```

```
address, blockquote, center, div, figure, figcaption, footer,
```

```
form, header, hr,
```

```
legend, listing, main, p, plaintext, pre, summary, xmp, article,
```

```
aside, h1, h2,
```

```
h3, h4, h5, h6, hgroup, nav, section, table, caption, colgroup,
```

```
col, thead,
```

```
tbody, tfoot, tr, td, th, dir, dd, dl, dt, menu, ol, ul, li, bdi,
```

```
output,
```

```
[dir=ltr i], [dir=rtl i], [dir=auto i] {
```

```
    unicode-bidi: isolate;
```

```
}
```

```
bdo, bdo[dir] { unicode-bidi: isolate-override; }
```

```
input[dir=auto i]:is([type=search i], [type=tel i], [type=url i],
```

```
[type=email i]), textarea[dir=auto i], pre[dir=auto i] {
```

```
    unicode-bidi: plaintext;
```

```
}
```

```
/* see prose for input elements whose type attribute is in the
```

```
Text state */
```

```
/* the rules setting the 'content' property on br and wbr
```

```
elements also has bidi implications */
```

Lorsque l'attribut `input` d'un élément `dir` est dans l'état [automatique](#) et que son `type` attribut est dans l'état [Texte](#), l'agent utilisateur est censé agir comme s'il avait une règle de feuille de style au niveau de l'agent utilisateur définissant la propriété '[unicode-bidi](#)' sur 'plaintext'.

Les champs de saisie (c'est-à-dire `textarea` les éléments et `input` les éléments lorsque leur `type` attribut est dans l'état [Text](#), [Search](#), [Telephone](#), [URL](#) ou [Email](#)) doivent présenter une interface utilisateur d'édition avec une directionnalité qui correspond à la propriété '[direction](#)' de l'élément.



Lorsque l'encodage des caractères du document est [ISO-8859-8](#) , les règles suivantes sont également censées s'appliquer, à la suite de celles ci-dessus : [\[ENCODING\]](#)

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
address, blockquote, center, div, figure, figcaption, footer,  
form, header, hr,
```

```
legend, listing, main, p, plaintext, pre, summary, xmp, article,  
aside, h1, h2,
```

```
h3, h4, h5, h6, hgroup, nav, section, table, caption, colgroup,  
col, thead,
```

```
tbody, tfoot, tr, td, th, dir, dd, dl, dt, menu, ol, ul, li,  
[dir=ltr i],
```

```
[dir=rtl i], [dir=auto i], *|* {
```

```
    unicode-bidi: bidi-override;
```

```
}
```

```
input:not([type=submit i]):not([type=reset i]):not([type=button  
i]),
```

```
textarea {
```

```
    unicode-bidi: normal;
```

```
}
```

### 15.3.6 Sections et titres

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
article, aside, h1, h2, h3, h4, h5, h6, hgroup, nav, section {
```

```
display: block;
```

```
}
```

```
h1 { margin-block-start: 0.67em; margin-block-end: 0.67em; font-size: 2.00em; font-weight: bold; }
```

```
h2 { margin-block-start: 0.83em; margin-block-end: 0.83em; font-size: 1.50em; font-weight: bold; }
```

```
h3 { margin-block-start: 1.00em; margin-block-end: 1.00em; font-size: 1.17em; font-weight: bold; }
```

```
h4 { margin-block-start: 1.33em; margin-block-end: 1.33em; font-size: 1.00em; font-weight: bold; }
```

```
h5 { margin-block-start: 1.67em; margin-block-end: 1.67em; font-size: 0.83em; font-weight: bold; }
```

```
h6 { margin-block-start: 2.33em; margin-block-end: 2.33em; font-size: 0.67em; font-weight: bold; }
```

Dans le bloc CSS suivant, x est un raccourci pour le sélecteur suivant : `:is(article, aside, nav, section)`

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
x h1 { margin-block-start: 0.83em; margin-block-end: 0.83em; font-size: 1.50em; }
```

```
x x h1 { margin-block-start: 1.00em; margin-block-end: 1.00em; font-size: 1.17em; }
```

```
x x x h1 { margin-block-start: 1.33em; margin-block-end: 1.33em;  
font-size: 1.00em; }
```

```
x x x x h1 { margin-block-start: 1.67em; margin-block-end:  
1.67em; font-size: 0.83em; }
```

```
x x x x x h1 { margin-block-start: 2.33em; margin-block-end:  
2.33em; font-size: 0.67em; }
```

*La sténographie est utilisée pour garder ce bloc au moins légèrement lisible.*

### 15.3.7 Listes

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
dir, dd, dl, dt, menu, ol, ul { display: block; }
```

```
li { display: list-item; text-align: match-parent; }
```

```
dir, dl, menu, ol, ul { margin-block-start: 1em; margin-block-  
end: 1em; }
```

```
:is(dir, dl, menu, ol, ul) :is(dir, dl, menu, ol, ul) {
```

```
margin-block-start: 0; margin-block-end: 0;
```

```
}
```

```
dd { margin-inline-start: 40px; }
```

```
dir, menu, ol, ul { padding-inline-start: 40px; }
```

```
ol, ul, menu { counter-reset: list-item; }
```

```
ol { list-style-type: decimal; }
```

```
dir, menu, ul {
```

```
list-style-type: disc;
```

```
}
```

```
:is(dir, menu, ol, ul) :is(dir, menu, ul) {
```

```
list-style-type: circle;
```

```
}
```

```
:is(dir, menu, ol, ul) :is(dir, menu, ol, ul) :is(dir, menu, ul)
```

```
{
```

```
list-style-type: square;
```

```
}
```

Les règles suivantes sont également censées s'appliquer, comme [conseils de présentation](#) :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
ol[type="1"], li[type="1"] { list-style-type: decimal; }
```

```
ol[type=a s], li[type=a s] { list-style-type: lower-alpha; }
```

```
ol[type=A s], li[type=A s] { list-style-type: upper-alpha; }
```

```
ol[type=i s], li[type=i s] { list-style-type: lower-roman; }
```

```
ol[type=I s], li[type=I s] { list-style-type: upper-roman; }
```

```
ul[type=none i], li[type=none i] { list-style-type: none; }
```

```
ul[type=disc i], li[type=disc i] { list-style-type: disc; }
```

```
ul[type=circle i], li[type=circle i] { list-style-type: circle; }
```

```
ul[type=square i], li[type=square i] { list-style-type: square; }
```

Lors du rendu des liéléments, les agents utilisateurs non-CSS sont censés utiliser la valeur ordinale de l' liélément pour rendre le compteur dans le marqueur d'élément de liste.

Pour les agents utilisateurs CSS, certains aspects du rendu des éléments de liste sont définis par la spécification *des listes CSS* . En outre, les mappages d'attributs suivants doivent s'appliquer : [CSSLISTS]

Lorsqu'un liélément a un valueattribut et que l'analyse de la valeur de cet attribut à l'aide des règles d'analyse des entiers ne génère pas d'erreur, l'agent utilisateur est censé utiliser la *valeur* de la valeur analysée comme indice de présentation pour la propriété 'counter-set' du forme `.list-item value`

Lorsqu'un olélément a un start attribut ou un reversedattribut, ou les deux, l'agent utilisateur est censé utiliser les étapes suivantes pour traiter les attributs comme une indication de présentation pour la propriété 'counter-reset' :

1. Soit *la valeur* nulle.
2. Si l'élément a un startattribut, définissez *valeur* sur le résultat de l'analyse de la valeur de l'attribut à l'aide des règles d'analyse des entiers .
3. Si l'élément a un reversedattribut, alors :
  1. Si *la valeur* est un entier, alors incrémentez *la valeur* de 1 et retournez `.reversed(list-item) value`
  2. Sinon, retournez `reversed(list-item)`.

*Soit l' startattribut était absent, soit l'analyse de sa valeur a généré une erreur.*
4. Sinon:
  1. Si *la valeur* est un entier, alors décrémente *la valeur* de 1 et retournez `.list-item value`
  2. Sinon, il n'y a pas d'indice de présentation .

### 15.3.8 Tableaux

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
table { display: table; }
```

```
caption { display: table-caption; }
```

```
colgroup, colgroup[hidden] { display: table-column-group; }
```

```
col, col[hidden] { display: table-column; }
```

```
thead, thead[hidden] { display: table-header-group; }
```

```
tbody, tbody[hidden] { display: table-row-group; }
```

```
tfoot, tfoot[hidden] { display: table-footer-group; }
```

```
tr, tr[hidden] { display: table-row; }
```

```
td, th { display: table-cell; }
```

```
colgroup[hidden], col[hidden], thead[hidden], tbody[hidden],
```

```
tfoot[hidden], tr[hidden] {
```

```
  visibility: collapse;
```

```
}
```

```
table {
```

```
  box-sizing: border-box;
```

```
  border-spacing: 2px;
```

```
  border-collapse: separate;
```

```
  text-indent: initial;
```

```
}
```

```
td, th { padding: 1px; }
```

```
th { font-weight: bold; }
```

```
caption { text-align: center; }
```

```
thead, tbody, tfoot, table > tr { vertical-align: middle; }
```

```
tr, td, th { vertical-align: inherit; }
```

```
thead, tbody, tfoot, tr { border-color: inherit; }
```

```
table[rules=none i], table[rules=groups i], table[rules=rows i],
```

```
table[rules=cols i], table[rules=all i], table[frame=void i],
```

```
table[frame=above i], table[frame=below i], table[frame=hsides
```

```
i],
```

```
table[frame=lhs i], table[frame=rhs i], table[frame=vsides i],
```

```
table[frame=box i], table[frame=border i],
```

```
table[rules=none i] > tr > td, table[rules=none i] > tr > th,
```

```
table[rules=groups i] > tr > td, table[rules=groups i] > tr > th,
```

```
table[rules=rows i] > tr > td, table[rules=rows i] > tr > th,
```

```
table[rules=cols i] > tr > td, table[rules=cols i] > tr > th,
```

```
table[rules=all i] > tr > td, table[rules=all i] > tr > th,
```

```
table[rules=none i] > thead > tr > td, table[rules=none i] >
```

```
thead > tr > th,
```

```
table[rules=groups i] > thead > tr > td, table[rules=groups i] >
```

```
thead > tr > th,
```

```
table[rules=rows i] > thead > tr > td, table[rules=rows i] >
```

```
thead > tr > th,
```

```
table[rules=cols i] > thead > tr > td, table[rules=cols i] >
```

```
thead > tr > th,
```

```
table[rules=all i] > thead > tr > td, table[rules=all i] > thead
```

```
> tr > th,
```

```
table[rules=none i] > tbody > tr > td, table[rules=none i] >
```

```
tbody > tr > th,
```

```
table[rules=groups i] > tbody > tr > td, table[rules=groups i] >
```

```
tbody > tr > th,
```

```
table[rules=rows i] > tbody > tr > td, table[rules=rows i] >
```

```
tbody > tr > th,
```

```
table[rules=cols i] > tbody > tr > td, table[rules=cols i] >
```

```
tbody > tr > th,
```

```
table[rules=all i] > tbody > tr > td, table[rules=all i] > tbody
```

```
> tr > th,
```

```
table[rules=none i] > tfoot > tr > td, table[rules=none i] >
```

```
tfoot > tr > th,
```

```
table[rules=groups i] > tfoot > tr > td, table[rules=groups i] >
```

```
tfoot > tr > th,
```

```
table[rules=rows i] > tfoot > tr > td, table[rules=rows i] >
```

```
tfoot > tr > th,
```

```
table[rules=cols i] > tfoot > tr > td, table[rules=cols i] >
```

```
tfoot > tr > th,
```

```
table[rules=all i] > tfoot > tr > td, table[rules=all i] > tfoot
```

```
> tr > th {
```

```
border-color: black;
```

```
}
```



Les règles suivantes sont également censées s'appliquer, comme [conseils de présentation](#) :

```
@namespace url(http://www.w3.org/1999/xhtmll);
```

```
table[align=left i] { float: left; }
```

```
table[align=right i] { float: right; }
```

```
table[align=center i] { margin-inline-start: auto; margin-inline-end: auto; }
```

```
thead[align=absmiddle i], tbody[align=absmiddle i],
```

```
tfoot[align=absmiddle i],
```

```
tr[align=absmiddle i], td[align=absmiddle i], th[align=absmiddle i] {
```

```
text-align: center;
```

```
}
```

```
caption[align=bottom i] { caption-side: bottom; }
```

```
p[align=left i], h1[align=left i], h2[align=left i],
```

```
h3[align=left i],
```

```
h4[align=left i], h5[align=left i], h6[align=left i] {
```

```
text-align: left;
```

```
}
```

```
p[align=right i], h1[align=right i], h2[align=right i],
```

```
h3[align=right i],
```

```
h4[align=right i], h5[align=right i], h6[align=right i] {
```

```
text-align: right;
```

```
}
```

```
p[align=center i], h1[align=center i], h2[align=center i],
```

```
h3[align=center i],
```

```
h4[align=center i], h5[align=center i], h6[align=center i] {
```

```
text-align: center;
```

```
}
```

```
p[align=justify i], h1[align=justify i], h2[align=justify i],
```

```
h3[align=justify i],
```

```
h4[align=justify i], h5[align=justify i], h6[align=justify i] {
```

```
text-align: justify;
```

```
}
```

```
thead[valign=top i], tbody[valign=top i], tfoot[valign=top i],
```

```
tr[valign=top i], td[valign=top i], th[valign=top i] {
```

```
vertical-align: top;
```

```
}
```

```
thead[valign=middle i], tbody[valign=middle i],
```

```
tfoot[valign=middle i],
```

```
tr[valign=middle i], td[valign=middle i], th[valign=middle i] {
```

```
vertical-align: middle;
```

```
}
```

```
thead[valign=bottom i], tbody[valign=bottom i],
```

```
tfoot[valign=bottom i],
```

```
tr[valign=bottom i], td[valign=bottom i], th[valign=bottom i] {
```

```
vertical-align: bottom;
```

```
}
```

```
thead[valign=baseline i], tbody[valign=baseline i],
```

```
tfoot[valign=baseline i],
```

```
tr[valign=baseline i], td[valign=baseline i], th[valign=baseline
```

```
i] {
```

```
vertical-align: baseline;
```

```
}
```

```
td[nowrap], th[nowrap] { white-space: nowrap; }
```

```
table[rules=none i], table[rules=groups i], table[rules=rows i],
```

```
table[rules=cols i], table[rules=all i] {
```

```
border-style: hidden;
```

```
border-collapse: collapse;
```

```
}
```

```
table[border] { border-style: outset; } /* only if border is not
```

```
equivalent to zero */
```

```
table[frame=void i] { border-style: hidden; }
```

```
table[frame=above i] { border-style: outset hidden hidden hidden;
```

```
}
```

```
table[frame=below i] { border-style: hidden hidden outset hidden;
```

```
}
```

```
table[frame=hsides i] { border-style: outset hidden outset  
hidden; }
```

```
table[frame=lhs i] { border-style: hidden hidden hidden outset; }
```

```
table[frame=rhs i] { border-style: hidden outset hidden hidden; }
```

```
table[frame=vsides i] { border-style: hidden outset; }
```

```
table[frame=box i], table[frame=border i] { border-style: outset;  
}
```

```
table[border] > tr > td, table[border] > tr > th,
```

```
table[border] > thead > tr > td, table[border] > thead > tr > th,
```

```
table[border] > tbody > tr > td, table[border] > tbody > tr > th,
```

```
table[border] > tfoot > tr > td, table[border] > tfoot > tr > th
```

```
{
```

```
/* only if border is not equivalent to zero */
```

```
border-width: 1px;
```

```
border-style: inset;
```

```
}
```

```
table[rules=none i] > tr > td, table[rules=none i] > tr > th,
```

```
table[rules=none i] > thead > tr > td, table[rules=none i] >  
thead > tr > th,
```

```
table[rules=none i] > tbody > tr > td, table[rules=none i] >  
tbody > tr > th,
```

```
table[rules=none i] > tfoot > tr > td, table[rules=none i] >  
tfoot > tr > th,
```

```
table[rules=groups i] > tr > td, table[rules=groups i] > tr > th,
```

```
table[rules=groups i] > thead > tr > td, table[rules=groups i] >
```

```
thead > tr > th,
```

```
table[rules=groups i] > tbody > tr > td, table[rules=groups i] >
```

```
tbody > tr > th,
```

```
table[rules=groups i] > tfoot > tr > td, table[rules=groups i] >
```

```
tfoot > tr > th,
```

```
table[rules=rows i] > tr > td, table[rules=rows i] > tr > th,
```

```
table[rules=rows i] > thead > tr > td, table[rules=rows i] >
```

```
thead > tr > th,
```

```
table[rules=rows i] > tbody > tr > td, table[rules=rows i] >
```

```
tbody > tr > th,
```

```
table[rules=rows i] > tfoot > tr > td, table[rules=rows i] >
```

```
tfoot > tr > th {
```

```
border-width: 1px;
```

```
border-style: none;
```

```
}
```

```
table[rules=cols i] > tr > td, table[rules=cols i] > tr > th,
```

```
table[rules=cols i] > thead > tr > td, table[rules=cols i] >
```

```
thead > tr > th,
```

```
table[rules=cols i] > tbody > tr > td, table[rules=cols i] >
```

```
tbody > tr > th,
```

```
table[rules=cols i] > tfoot > tr > td, table[rules=cols i] >
```

```
tfoot > tr > th {
```

```
border-width: 1px;
```

```
border-block-start-style: none;
```

```
border-inline-end-style: solid;
```

```
border-block-end-style: none;
```

```
border-inline-start-style: solid;
```

```
}
```

```
table[rules=all i] > tr > td, table[rules=all i] > tr > th,
```

```
table[rules=all i] > thead > tr > td, table[rules=all i] > thead
```

```
> tr > th,
```

```
table[rules=all i] > tbody > tr > td, table[rules=all i] > tbody
```

```
> tr > th,
```

```
table[rules=all i] > tfoot > tr > td, table[rules=all i] > tfoot
```

```
> tr > th {
```

```
border-width: 1px;
```

```
border-style: solid;
```

```
}
```

```
table[rules=groups i] > colgroup {
```

```
border-inline-start-width: 1px;
```

```
border-inline-start-style: solid;
```

```
border-inline-end-width: 1px;
```

```
border-inline-end-style: solid;
```

```
}
```

```
table[rules=groups i] > thead,
```

```
table[rules=groups i] > tbody,
```

```
table[rules=groups i] > tfoot {
```

```
border-block-start-width: 1px;
```

```
border-block-start-style: solid;
```

```
border-block-end-width: 1px;
```

```
border-block-end-style: solid;
```

```
}
```

```
table[rules=rows i] > tr, table[rules=rows i] > thead > tr,
```

```
table[rules=rows i] > tbody > tr, table[rules=rows i] > tfoot >
```

```
tr {
```

```
border-block-start-width: 1px;
```

```
border-block-start-style: solid;
```

```
border-block-end-width: 1px;
```

```
border-block-end-style: solid;
```

```
}
```

En [mode quirks](#) , les règles suivantes sont également censées s'appliquer :

```
@namespace url (http://www.w3.org/1999/xhtml);
```

```
table {
```

```
font-weight: initial;
```

```
font-style: initial;
```

```
font-variant: initial;
```

```
font-size: initial;
```

```
line-height: initial;
```

```
white-space: initial;
```

```
text-align: initial;
```

```
}
```

---

Pour les besoins du modèle de table CSS, l'[col](#)élément doit être traité comme s'il était présent autant de fois que son [span](#)attribut [le spécifie](#) .

Pour les besoins du modèle de table CSS, l'[colgroup](#)élément, s'il ne contient aucun [col](#)élément, doit être traité comme s'il avait autant d'enfants que son [span](#)attribut [le spécifie](#) .

Pour les besoins du modèle de tableau CSS, les attributs [colspan](#)et [rowspan](#)sur les éléments [td](#)et [th](#) sont censés [fournir](#) les *connaissances particulières* concernant les cellules s'étendant sur des lignes et des colonnes.

Dans [les documents HTML](#) , les règles suivantes doivent également s'appliquer :

```
@namespace url (http://www.w3.org/1999/xhtml);
```

```
:is(table, thead, tbody, tfoot, tr) > form { display: none
```

```
!important; }
```

---

L'attribut [table](#)de l'élément [correspond à la propriété de longueur de pixel 'border-spacing'](#) sur l'élément.[cellspacing](#)



L'attribut `table` de l'élément `correspond` aux propriétés de longueur de pixel '`padding-top`', '`padding-right`', '`padding-bottom`' et '`padding-left`' de tous les éléments et qui ont `des cellules` correspondantes dans le `tableau` correspondant à l'élément. `cellpadding` `tdthtable`

L'attribut `table` de l'élément `correspond` à la propriété de dimension (sans tenir compte du zéro) '`height`' sur l'élément. `height` `table`

L'attribut `table` de l'élément `est mappé` à la propriété de dimension (ignorant zéro) '`width`' sur l'élément. `width` `table`

L'attribut `col` de l'élément `correspond` à la propriété de dimension '`width`' sur l'élément. `width` `col`

Les attributs des `thead` éléments `tbody`, et `tfoot` des éléments `correspondent` à la propriété de dimension '`hauteur`' de l'élément. `height`

L'attribut `tr` de l'élément `correspond` à la propriété de dimension '`height`' sur l'élément. `height` `tr`

Les attributs des éléments `td` et `correspondent` à la propriété dimension (en ignorant zéro) '`hauteur`' sur l'élément. `thheight`

Les attributs des éléments `td` et `correspondent` à la propriété de dimension (en ignorant zéro) '`width`' sur l'élément. `thwidth`

---

Les éléments `thead`, `tbody`, `tfoot`, `tr`, `td` et `th`, lorsqu'ils ont un `align` attribut dont la valeur est une correspondance `ASCII insensible à la casse` pour la chaîne "`center`" ou la chaîne "`middle`", sont censés centrer le texte sur eux-mêmes, comme s'ils avaient leur '`texte-align`' définie sur '`center`' dans un `indice de présentation` et pour `aligner les descendants` au centre.

Les éléments `thead`, `tbody`, `tfoot`, `tr`, `td` et `th`, lorsqu'ils ont un `align` attribut dont la valeur est une `correspondance ASCII insensible à la casse` pour la chaîne "`left`", sont censés aligner le texte à gauche sur eux-mêmes, comme s'ils avaient leur propriété '`text-align`' défini sur '`left`' dans un `indice de présentation` et pour `aligner les descendants` à gauche.

Les éléments `thead`, `tbody`, `tfoot`, `tr`, `td` et `th`, lorsqu'ils ont un `align` attribut dont la valeur est une correspondance `ASCII insensible à la casse` pour la chaîne "`right`", sont censés aligner le texte à droite sur eux-mêmes, comme s'ils avaient leur propriété '`text-align`' défini sur '`right`' dans un `indice de présentation` et pour `aligner les descendants` à droite.

Les éléments `thead`, `tbody`, `tfoot`, `tr`, `td` et `th`, lorsqu'ils ont un `align` attribut dont la valeur est une [correspondance ASCII insensible à la casse](#) pour la chaîne "justify", sont censés justifier entièrement le texte en eux-mêmes, comme s'ils avaient leur propriété `'text-align'` défini sur 'justify' dans un [indice de présentation](#) et pour [aligner les descendants](#) à gauche.

Les agents utilisateurs sont censés avoir une règle dans leur feuille de style d'agent utilisateur qui correspond aux `th`éléments qui ont un nœud parent dont [la valeur calculée](#) pour la propriété `'text-align'` est sa valeur initiale, dont le bloc de déclaration se compose d'une seule déclaration qui définit le propriété `'text-align'` à la valeur `'center'`.

---

Lorsqu'un élément `table`, `thead`, `tbody`, `tfoot`, `tr`, `td` ou `th` a un `background` attribut défini sur une valeur non vide, la nouvelle valeur est censée être [analysée](#) par rapport au [nœud document](#) de l'élément, et si cela réussit, l'agent utilisateur doit traiter le comme indice [de présentation](#) définissant la propriété `'background-image'` de l'élément sur la [chaîne d'URL résultante](#).

Lorsqu'un élément `table`, `thead`, `tbody`, `tfoot`, `tr`, `td` ou `th` a un `bgcolor` ensemble d'attributs, la nouvelle valeur doit être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#), et si cela ne renvoie pas d'erreur, l'agent utilisateur doit traiter le comme indice [de présentation](#) définissant la propriété `'background-color'` de l'élément sur la couleur résultante.

Lorsqu'un `table`élément a un `bordercolor` attribut, sa valeur est censée être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#), et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation définissant le](#) ' de l'élément `border-top-color`, `border-right-color`, `border-bottom-color` et `border-left-color` à la couleur résultante.

---

L'attribut `table` de l'élément [correspond aux propriétés de longueur de pixel](#) `'border-top-width'`, `'border-right-width'`, `'border-bottom-width'`, `'border-left-width'` sur l'élément. Si l'attribut est présent mais que l'analyse de la valeur de l'attribut à l'aide des [règles d'analyse des entiers non négatifs](#) génère une erreur, une valeur par défaut de 1px devrait être utilisée pour cette propriété à la place. `border`

Les règles marquées " **uniquement si la bordure n'est pas équivalente à zéro** " dans le bloc CSS ci-dessus ne devraient être appliquées que si l' `border` attribut mentionné dans les sélecteurs de la règle est non seulement présent mais, lorsqu'il

est analysé à l'aide des [règles d'analyse des entiers non négatifs](#) , s'avère également avoir une valeur autre que zéro ou générer une erreur.

---

En [mode quirks](#) , un [td](#)élément ou un [th](#)élément qui a un [nowrap](#)attribut mais qui a aussi un [width](#)attribut dont la valeur, lorsqu'elle est analysée à l'aide des [règles d'analyse des valeurs de dimension non nulles](#) , s'avère être une longueur (et non une erreur ou un nombre classé en pourcentage) , devrait avoir un [indice de présentation](#) définissant la propriété ['white-space' de l'élément sur 'normal'](#), remplaçant la règle dans le bloc CSS ci-dessus qui le définit sur ['nowrap'](#).

### 15.3.9 bizarreries d'effondrement de marge

Un nœud est **substantiel** s'il s'agit d'un nœud de texte qui n'est pas [un espace blanc inter-éléments](#) , ou s'il s'agit d'un nœud d'élément.

Un nœud est **vide** s'il s'agit d'un élément qui ne contient aucun nœud [substantiel](#) .

Les **éléments avec des marges par défaut** sont les éléments suivants

: [blockquote](#), [dir](#), [dl](#), [h1](#), [h2](#), [h3](#), [h4](#), [h5](#),  
[h6](#), [listing](#), [menu](#), [ol](#), [p](#), [plaintext](#), [pre](#),[ulxmp](#)

En [mode quirks](#) , tout [élément avec des marges par défaut](#) qui est l' [enfant](#) d'un élément [body](#), [td](#), ou [th](#)et qui n'a pas de frères et sœurs précédents [substantiels](#) devrait avoir une règle de feuille de style au niveau de l'agent utilisateur qui définit sa propriété ['margin-block-start'](#) à zéro .

En [mode quirks](#) , tout [élément avec des marges par défaut](#) qui est l' [enfant](#) d'un élément [body](#), [td](#), ou [th](#), n'a pas de frères et sœurs précédents [substantiels](#) et est [vide](#) , devrait avoir une règle de feuille de style au niveau de l'agent utilisateur qui définit son ['margin-block-end'](#) à zéro également.

En [mode quirks](#) , tout [élément avec des marges par défaut](#) qui est l' [enfant](#) d'un élément [td](#)ou [th](#), n'a pas de frères et sœurs suivants [substantiels](#) et est [vide](#) , devrait avoir une règle de feuille de style au niveau de l'agent utilisateur qui définit son ['margin-block-start'](#) propriété à zéro.

En [mode quirks](#) , tout [p](#)élément qui est l' [enfant](#) d'un élément [td](#)ou [th](#)et qui n'a pas de frères et sœurs [substantiels](#) , devrait avoir une règle de feuille de style au niveau de l'agent utilisateur qui définit sa propriété ['margin-block-end'](#) sur zéro.

### 15.3.10 Champs de formulaire

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
input, select, button, textarea {
```

```
  letter-spacing: initial;
```

```
  word-spacing: initial;
```

```
  line-height: initial;
```

```
  text-transform: initial;
```

```
  text-indent: initial;
```

```
  text-shadow: initial;
```

```
  appearance: auto;
```

```
}
```

```
input, select, textarea {
```

```
  text-align: initial;
```

```
}
```

```
input:is([type=reset i], [type=button i], [type=submit i]),
```

```
button {
```

```
  text-align: center;
```

```
}
```

```
input, button {
```

```
  display: inline-block;
```

```
}
```

```
input[type=hidden i], input[type=file i], input[type=image i] {
```

```
  appearance: none;
```

```
}
```

```
input:is([type=radio i], [type=checkbox i], [type=reset i],
```

```
[type=button i],
```

```
[type=submit i], [type=button i], [type=search i]), select, button
```

```
{
```

```
  box-sizing: border-box;
```

```
}
```

```
textarea { white-space: pre-wrap; }
```

En [mode quirks](#) , les règles suivantes sont également censées s'appliquer :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
input:not([type=image i]), textarea { box-sizing: border-box; }
```

Chaque type de contrôle de formulaire est également décrit dans la section [Widgets](#) , qui décrit l'apparence du contrôle.

Pour [input](#) les éléments où l' [type](#) attribut n'est pas dans l' état [Masqué](#) ou dans l' état [Bouton d'image](#) , et qui sont [rendus](#) , ils doivent agir comme suit :

- Le [type d'affichage interne](#) est toujours 'flow-root'.
- La propriété ['overflow'](#) est ignorée et se comporte toujours comme 'visible' dans le but d'interagir avec d'autres fonctionnalités CSS (en particulier, la propriété ['vertical-align'](#) ), mais coupe toujours tout débordement au bord de la bordure, et pas de défilement mécanique s'affiche.

### 15.3.11 L' hr élément

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
hr {
```

```
  color: gray;
```

```
  border-style: inset;
```

```
  border-width: 1px;
```

```
  margin-block-start: 0.5em;
```

```
  margin-inline-end: auto;
```

```
  margin-block-end: 0.5em;
```

```
  margin-inline-start: auto;
```

```
  overflow: hidden;
```

```
}
```

Les règles suivantes sont également censées s'appliquer, comme [conseils de présentation](#) :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
hr[align=left i] { margin-left: 0; margin-right: auto; }
```

```
hr[align=right i] { margin-left: auto; margin-right: 0; }
```

```
hr[align=center i] { margin-left: auto; margin-right: auto; }
```

```
hr[color], hr[noshade] { border-style: solid; }
```

Si un hr élément a soit un color attribut, soit un noshade attribut, et en outre a également un size attribut, et que l'analyse de la valeur de cet attribut à l'aide des [règles d'analyse des entiers non négatifs](#) ne génère pas d'erreur, alors l'agent utilisateur est censé utiliser la valeur analysée divisé par deux comme une longueur de pixel pour [les conseils de présentation](#) pour les propriétés '[border-top-width](#)', '[border-right-width](#)', '[border-bottom-width](#)' et '[border-left-width](#)' sur l'élément.

Sinon, si un hrélément n'a ni color attribut ni noshadeattribut, mais qu'il a un sizeattribut, et que l'analyse de la valeur de cet attribut à l'aide des règles d'analyse des entiers non négatifs ne génère pas d'erreur, alors : si la valeur analysée est un, alors on s'attend à ce que l'agent utilisateur utilise l'attribut comme indice de présentation en fixant la valeur 'border-bottom-width' de l'élément à 0 ; sinon, si la valeur analysée est supérieure à un, alors l'agent utilisateur est censé utiliser la valeur analysée moins deux comme longueur de pixel pour les conseils de présentation pour la propriété 'height' sur l'élément.

L' widthattribut d'un hrélément correspond à la propriété de dimension 'width' de l'élément.

Lorsqu'un hrélément a un colorattribut, sa valeur est censée être analysée en utilisant les règles d'analyse d'une valeur de couleur héritée , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un indice de présentation définissant le ' de l'élément color' à la couleur résultante.

### 15.3.12 Les éléments fieldsetetlegend

```
@namespace url (http://www.w3.org/1999/xhtml);
```

```
fieldset {
```

```
display: block;
```

```
margin-inline-start: 2px;
```

```
margin-inline-end: 2px;
```

```
border: groove 2px ThreeDFace;
```

```
padding-block-start: 0.35em;
```

```
padding-inline-end: 0.75em;
```

```
padding-block-end: 0.625em;
```

```
padding-inline-start: 0.75em;
```

```
min-inline-size: min-content;
```

```
}
```

```
legend {
```

```
padding-inline-start: 2px; padding-inline-end: 2px;
```

```
}
```

```
legend[align=left i] {
```

```
justify-self: left;
```

```
}
```

```
legend[align=center i] {
```

```
justify-self: center;
```

```
}
```

```
legend[align=right i] {
```

```
justify-self: right;
```

```
}
```

L' fieldset élément, lorsqu'il génère une boîte CSS , doit agir comme suit :

- L'élément est censé établir un nouveau contexte de formatage de bloc .
- La propriété 'display' devrait agir comme suit :
  - Si la valeur calculée de 'display' est une valeur telle que le type d'affichage externe est 'inline', alors se comporter comme 'inline-block'.
  - Sinon, comportez-vous comme 'flow-root'.

*Cela ne change pas la valeur calculée.*

- **Si la boîte de l'élément a une boîte enfant qui correspond aux conditions de la liste ci-dessous, alors la première de ces boîtes enfant est la légende rendue de l'élément 'fieldset' :**
  - L'enfant est un legend élément.



- La valeur utilisée par l'enfant de '[float](#)' est 'none'.
  - La valeur utilisée par l'enfant de '[position](#)' n'est pas 'absolue' ou 'fixe'.
- Si l'élément a une [légende rendue](#) , alors la bordure ne devrait pas être peinte derrière le rectangle défini comme suit, en utilisant le mode d'écriture du fieldset :
  - Le bord de début de bloc du rectangle est le plus petit entre le bord de début de bloc du rectangle de marge de la [légende rendue](#) à sa position statique (sans tenir compte des transformations) et le bord extérieur de début de bloc de la [fieldset](#) bordure de .
  - Le bord de fin de bloc du rectangle est le plus grand entre le bord de fin de bloc du rectangle de marge de [la légende rendue](#) à sa position statique (en ignorant les transformations) et le bord extérieur de fin de bloc de la [fieldset](#) bordure de .
  - Le bord de début en ligne du rectangle est le plus petit entre le bord de début en ligne du rectangle de bordure de la [légende rendue](#) à sa position statique (en ignorant les transformations) et le bord extérieur de début en ligne de la [fieldset](#) bordure de .
  - Le bord de fin de ligne du rectangle est le plus grand entre le bord de fin de ligne du rectangle de bordure de [la légende rendue](#) à sa position statique (en ignorant les transformations) et le bord extérieur de fin de ligne de la [fieldset](#) bordure de .
- L'espace alloué pour la bordure de l'élément du côté du début du bloc est censé être la [largeur de la bordure du bloc de](#) l'élément ou la taille de la boîte de marge de la [légende rendue](#)[fieldset](#) dans la direction du flux du bloc, selon la valeur la plus élevée.
- Aux fins du calcul de la '[taille de bloc](#)' utilisée, si la '[taille de bloc](#)' calculée n'est pas 'auto', l'espace alloué pour la boîte de marge de la [légende rendue](#) qui déborde au-delà de la bordure, le cas échéant, est attendu à soustraire de la '[block-size](#)' . Si la taille de bloc de la boîte de contenu est négative, laissez la taille de bloc de la boîte de contenu à zéro à la place.
- Si l'élément a une [légende rendue](#) , cet élément est censé être la première boîte enfant.
- La [zone de contenu de l'ensemble de champs anonymes](#) devrait apparaître après la [légende rendue](#) et devrait contenir le contenu (y compris les pseudo-éléments '::before' et '::after') de l' [fieldset](#) élément à l'exception de la [légende rendue](#) , s'il y a un.
- La valeur utilisée des propriétés '[padding-top](#)' , '[padding-right](#)' , '[padding-bottom](#)' et '[padding-left](#)' devrait être nulle.

- Aux fins du calcul de la taille min-content inline, utilisez la valeur la plus élevée entre la taille min-content inline de la [légende rendue](#) et la taille min-content inline de la [zone de contenu de l'ensemble de champs anonymes](#) .
- Aux fins du calcul de la taille max-content inline, utilisez la plus grande de la taille max-content inline de la [légende rendue](#) et de la taille max-content inline de la [zone de contenu de l'ensemble de champs anonymes](#) .

La [légende rendue](#)[fieldset](#) d'un élément , le cas échéant, doit agir comme suit :

- L'élément est censé établir un nouveau [contexte de formatage](#) pour son contenu. Le type de ce [contexte de formatage](#) est déterminé par sa valeur '[display](#)' , comme d'habitude.
- La propriété '[display](#)' est censée se comporter comme si sa valeur calculée était bloquée.

*Cela ne change pas la valeur calculée.*

- Si la [valeur calculée](#) de '[inline-size](#)' est 'auto', alors la [valeur utilisée](#) est la [fit-content inline size](#) .
- L'élément doit être positionné dans la direction en ligne, comme c'est normal pour les blocs (par exemple, en tenant compte des marges et de la propriété '[justify-self](#)' ).
- On s'attend à ce que la boîte de l'élément soit contrainte dans la direction en ligne par la taille du contenu en ligne de comme [fieldset](#) s'il avait utilisé son remplissage en ligne calculé.

Par exemple, si le [fieldset](#) a un rembourrage spécifié de 50 pixels, la [légende rendue](#) sera positionnée à 50 pixels de la [fieldset](#) bordure de . Le rembourrage s'appliquera également à la [zone de contenu de l'ensemble de champs anonymes](#) au lieu de l' [fieldset](#) élément lui-même.

- L'élément est censé être positionné dans la direction du flux de blocs de sorte que sa boîte de bordure soit centrée sur la bordure du côté de début de bloc de l' [fieldset](#) élément.

La **zone de contenu de l'ensemble de champs anonyme**[fieldset](#) d'un élément doit agir comme suit :

- La propriété '[display](#)' devrait agir comme suit :
  - Si la valeur calculée de '[display](#)' sur l' [fieldset](#) élément est 'grid' ou 'inline-grid', alors définissez la valeur utilisée sur 'grid'.
  - Si la valeur calculée de '[display](#)' sur l' [fieldset](#) élément est 'flex' ou 'inline-flex', alors définissez la valeur utilisée sur 'flex'.
  - Sinon, définissez la valeur utilisée sur 'flow-root'.

- Les propriétés suivantes devraient hériter de l' [fieldset](#) élément :
  - ['aligner le contenu'](#)
  - ['aligner les éléments'](#)
  - ['rayon de bordure'](#)
  - ['nombre de colonnes'](#)
  - ['remplir la colonne'](#)
  - ['espace de colonne'](#)
  - ['règle de colonne'](#)
  - ['largeur de colonne'](#)
  - ['flex-direction'](#)
  - ['flex-wrap'](#)
  - ['grille-auto-colonnes'](#)
  - ['grid-flux automatique'](#)
  - ['grid-auto-lignes'](#)
  - ['gap-colonne-gap'](#)
  - ['écart de ligne de grille'](#)
  - ['grid-template-areas'](#)
  - ['grille-modèle-colonnes'](#)
  - ['grille-modèle-lignes'](#)
  - ['justifier le contenu'](#)
  - ['justifier les éléments'](#)
  - ['débordement'](#)
  - ['bas de rembourrage'](#)
  - ['padding-gauche'](#)
  - ['remplissage à droite'](#)
  - ['rembourrage haut'](#)
  - ['débordement de texte'](#)
  - ['unicode-bidi'](#)
- La propriété ['block-size'](#) devrait être définie sur '100%'.
- Aux fins du calcul du pourcentage de rembourrage, agissez comme si le rembourrage était calculé pour l' [fieldset](#) élément.

*fieldset's marginlegendpaddinglegend's marginpaddinganonymous fieldset content boxcontent*  
 La légende est rendue sur la bordure supérieure et la zone de bordure supérieure réserve un espace vertical pour la légende. La marge supérieure du jeu de champs commence au bord de la marge supérieure de la légende. Les marges horizontales de la légende, ou la propriété ['justify-self'](#) , donne sa position horizontale. La [zone de contenu de l'ensemble de champs anonyme](#) apparaît sous la légende.

## 15.4 Éléments remplacés

Les éléments suivants peuvent être [des éléments remplacés](#) : [audio](#), [canvas](#), [embed](#), [iframe](#), [img](#), [input](#), [object](#), et [video](#).

### 15.4.1 Contenu intégré

Les éléments `embed`, `iframe` et `video` doivent être traités comme [des éléments remplacés](#) .

Un `canvas` élément qui [représente un contenu intégré](#) est censé être traité comme un [élément remplacé](#) ; le contenu de tels éléments est le bitmap de l'élément, le cas échéant, ou bien un bitmap [noir transparent avec les mêmes dimensions intrinsèques](#) que l'élément. Les autres `canvas` éléments sont censés être traités comme des éléments ordinaires dans le modèle de rendu.

Un `object` élément qui [représente](#) une image, un plug-in ou son [contenu navigable](#) doit être traité comme un [élément remplacé](#) . Les autres `object` éléments sont censés être traités comme des éléments ordinaires dans le modèle de rendu.

L' `audio` élément, lorsqu'il [expose une interface utilisateur](#) , est censé être traité comme un [élément remplacé d'](#) une hauteur d'environ une ligne, aussi large que nécessaire pour exposer les fonctionnalités de l'interface utilisateur de l'agent utilisateur. Lorsqu'un `audio` élément n'expose pas [une interface utilisateur](#) , l'agent utilisateur doit forcer sa propriété `'display'` à calculer à `'none'`, [quelles que soient les règles CSS](#).

Le fait qu'un `video` élément [expose une interface utilisateur](#) ne devrait pas affecter la taille du rendu ; les contrôles doivent être superposés au-dessus du contenu de la page sans entraîner de modifications de mise en page et doivent disparaître lorsque l'utilisateur n'en a pas besoin.

Lorsqu'un `video` élément représente une image d'affiche ou une image de vidéo, l'image d'affiche ou l'image de vidéo doit être rendue à la plus grande taille qui conserve le rapport d'aspect de cette image d'affiche ou de cette image de vidéo sans être plus haute ou plus large que l'élément lui- `video` même , et devrait être centré dans l' `video` élément.

Tous les sous-titres ou légendes doivent être superposés directement au-dessus de leur `video` élément, tel que défini par les règles de rendu pertinentes ; pour WebVTT, ce sont les [règles de mise à jour de l'affichage des pistes de texte WebVTT](#) . [\[WEBVTT\]](#)

Lorsque l'agent utilisateur commence [à exposer une interface utilisateur](#) pour un `video` élément, l'agent utilisateur doit exécuter les [règles de mise à jour du rendu de la piste de texte](#) de chacune des [pistes de texte](#) dans la [liste des pistes de texte](#) `video` de l'élément qui [s'affichent](#) et dont [le type de piste de texte](#) est l'un des ou (par exemple, pour [les pistes de texte](#) basées sur WebVTT, les [règles de mise à jour de l'affichage des pistes de texte WebVTT](#) ). [\[WEBVTT\]](#) `subtitlescaptions`

*Le redimensionnement `video` et `canvas` les éléments n'interrompent pas la lecture vidéo et n'effacent pas le canevas.*

---

Les règles CSS suivantes devraient s'appliquer :

```
@namespace url (http://www.w3.org/1999/xhtml);
```

```
iframe { border: 2px inset; }
```

```
video { object-fit: contain; }
```

## 15.4.2 Images

Les agents utilisateurs sont censés restituer img les éléments et input les éléments dont type les attributs sont dans l'état [Image Button](#) , selon les premières règles applicables de la liste suivante :

### Si l'élément [représente](#) une image

L'agent utilisateur est censé traiter l'élément comme un [élément remplacé](#) et restituer l'image conformément aux règles définies dans CSS.

### Si l'élément ne [représente](#) pas une image et soit :

- l'agent utilisateur a des raisons de croire que l'image deviendra [disponible](#) et sera rendue en temps voulu, ou
- l'élément n'a pas `alt` d'attribut, ou
- le Document est en [mode Quirks](#) et l'élément a déjà [des dimensions intrinsèques](#) (par exemple, à partir des [attributs de dimension](#) ou des règles CSS)

L'agent utilisateur est censé traiter l'élément comme un [élément remplacé](#) dont le contenu est le texte que l'élément représente, le cas échéant, éventuellement accompagné d'une icône indiquant que l'image est en cours d'obtention (le cas échéant). Pour input les éléments, l'élément doit apparaître comme un bouton pour indiquer que l'élément est un [bouton](#) .

### Si l'élément est un img élément qui [représente](#) du texte et que l'agent utilisateur ne s'attend pas à ce que cela change

L'agent utilisateur est censé traiter l'élément comme un élément de phrasé non remplacé dont le contenu est le texte, éventuellement avec une icône indiquant qu'une image est manquante, afin que l'utilisateur puisse demander l'affichage de l'image ou rechercher pourquoi elle n'est pas rendue . Dans les contextes non graphiques, une telle icône doit être omise.

Si l'élément est un imgélément qui ne représente rien et que l'agent utilisateur ne s'attend pas à ce que cela change

L'agent utilisateur est censé traiter l'élément comme un élément remplacé dont les dimensions intrinsèques sont 0. (En l'absence d'autres styles, l'élément ne sera pratiquement pas rendu.)

Si l'élément est un inputélément qui ne représente pas une image et que l'agent utilisateur ne s'attend pas à ce que cela change

L'agent utilisateur est censé traiter l'élément comme un élément remplacé consistant en un bouton dont le contenu est le texte alternatif de l'élément. Les dimensions intrinsèques du bouton devraient être d'environ une ligne de hauteur et quelle que soit la largeur nécessaire pour afficher le texte sur une ligne.

Les icônes mentionnées ci-dessus doivent être relativement petites afin de ne pas perturber la plupart du texte mais d'être facilement cliquables. Dans un environnement visuel, par exemple, les icônes peuvent faire 16 pixels sur 16 pixels carrés, ou 1 em sur 1 em si les images sont évolutives. Dans un environnement audio, l'icône peut être un bip court. Les icônes sont destinées à indiquer à l'utilisateur qu'elles peuvent être utilisées pour accéder à toutes les options fournies par l'UA pour les images et, le cas échéant, sont censées fournir un accès au menu contextuel qui serait apparu si l'utilisateur avait interagi avec le image réelle.

---

Toutes les images animées avec la même URL absolue et les mêmes données d'image doivent être rendues synchronisées sur la même chronologie en tant que groupe, la chronologie commençant au moment de l'ajout le moins récent au groupe.

*En d'autres termes, lorsqu'une deuxième image avec la même URL absolue et les mêmes données d'image animée est insérée dans un document, elle saute au point du cycle d'animation actuellement affiché par la première image.*

Lorsqu'un agent utilisateur doit **redémarrer l'animation** d'un imgélément affichant une image animée, toutes les images animées avec la même URL absolue et les mêmes données d'image dans le document de nœud de cet élément doivent redémarrer leur animation depuis le début.

---

Les règles CSS suivantes sont censées s'appliquer lorsque le Document est en mode Quirks :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
img[align=left i] { margin-right: 3px; }
```

```
img[align=right i] { margin-left: 3px; }
```

### 15.4.3 Attributs pour le contenu intégré et les images

Les règles CSS suivantes sont censées s'appliquer en tant [qu'astuces de présentation](#) :

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
iframe[frameborder='0'], iframe[frameborder=no i] { border: none;
```

```
}
```

```
embed[align=left i], iframe[align=left i], img[align=left i],
```

```
input[type=image i][align=left i], object[align=left i] {
```

```
float: left;
```

```
}
```

```
embed[align=right i], iframe[align=right i], img[align=right i],
```

```
input[type=image i][align=right i], object[align=right i] {
```

```
float: right;
```

```
}
```

```
embed[align=top i], iframe[align=top i], img[align=top i],
```

```
input[type=image i][align=top i], object[align=top i] {
```

```
vertical-align: top;
```

```
}
```

```
embed[align=baseline i], iframe[align=baseline i],
```

```
img[align=baseline i],
```

```
input[type=image i][align=baseline i], object[align=baseline i] {
```

```
vertical-align: baseline;
```

```
}
```

```
embed[align=texttop i], iframe[align=texttop i],
```

```
img[align=texttop i],
```

```
input[type=image i][align=texttop i], object[align=texttop i] {
```

```
vertical-align: text-top;
```

```
}
```

```
embed[align=absmiddle i], iframe[align=absmiddle i],
```

```
img[align=absmiddle i],
```

```
input[type=image i][align=absmiddle i], object[align=absmiddle
```

```
i],
```

```
embed[align=abscenter i], iframe[align=abscenter i],
```

```
img[align=abscenter i],
```

```
input[type=image i][align=abscenter i], object[align=abscenter i]
```

```
{
```

```
vertical-align: middle;
```

```
}
```



```

embed[align=bottom i], iframe[align=bottom i], img[align=bottom
i],
input[type=image i][align=bottom i], object[align=bottom i] {
vertical-align: bottom;
}

```

Lorsqu'un élément `embed`, `iframe`, `img` ou `object`, ou un `input` élément dont `type` l'attribut est dans l'état [Image Button](#), a un `align` attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne "center" ou la chaîne "middle", l'agent utilisateur est censé agir comme si la propriété '[vertical-align](#)' de l'élément était définie sur une valeur qui aligne le milieu vertical de l'élément avec la ligne de base de l'élément parent.

L' `hspace` attribut des éléments `embed`, `img`, ou `object`, et `input` les éléments avec un `type` attribut dans l'état [Image Button](#), [correspondent aux propriétés de dimension 'margin-left' et 'margin-right'](#) sur l'élément.

L' `vspace` attribut des éléments `embed`, `img`, ou `object`, et `input` les éléments avec un `type` attribut dans l'état [Image Button](#), [correspondent aux propriétés de dimension 'margin-top' et 'margin-bottom'](#) sur l'élément.

Lorsqu'un `img` élément, `object` un élément ou `input` un élément avec un `type` attribut dans l'état [Image Button](#) a un `border` attribut dont la valeur, lorsqu'elle est analysée à l'aide des [règles d'analyse des entiers non négatifs](#), s'avère être un nombre supérieur à zéro, l'agent utilisateur est censé utiliser la valeur analysée pour huit [conseils de présentation](#) : quatre définissant la valeur analysée comme une longueur de pixel pour les éléments '[border-top-width](#)', '[border-right-width](#)', '[border-bottom-width](#)' et '[border-left-width](#)', et quatre définissant le '[border-top-style](#)' de l'élément, '[border-right-style](#)', '[border-bottom-style](#)' et '[border-left-style](#)' à la valeur 'solid'.

Les attributs `width` et de [la source d'attributs de dimension](#) d'un élément correspondent respectivement [aux propriétés de dimension 'width' et 'height'](#) de l'élément. Ils sont également [mappés à la propriété de rapport d'aspect \(à l'aide de règles de dimension\)](#) de l'élément. `heightimg`

Les attributs `width` et `height` sur les éléments `embed`, `iframe`, `object` et `video`, et `input` les éléments avec un `type` attribut dans l'état [Image Button](#) et qui représentent une image ou dont l'utilisateur s'attend à ce qu'ils représentent éventuellement une image, [correspondent aux propriétés de dimension 'width' et 'height'](#) sur respectivement l'élément.

Les attributs `width` et [sont mappés à la propriété de rapport d'aspect \(à l'aide des règles de dimension\)](#) sur les éléments et les éléments avec un attribut dans l'état [Bouton d'image](#) `.heightimgvideoinputtype`

Les attributs `width` et [correspondent à la propriété de rapport d'aspect](#) sur les éléments `.heightcanvas`

#### 15.4.4 Images cliquables

Les formes sur une [image cliquable](#) sont censées agir, dans le cadre de la cascade CSS, comme des éléments indépendants de l' `area` élément d'origine qui correspondent aux mêmes règles de style mais héritent de l' élément `img` ou `.object`

Pour les besoins du rendu, seule la propriété `'cursor'` devrait avoir un effet sur la forme.

Ainsi, par exemple, si un `area` élément a un `style` attribut qui définit la propriété `'cursor'` sur 'help', alors lorsque l'utilisateur désigne cette forme, le curseur se transforme en curseur d'aide.

De même, si un `area` élément avait une règle CSS qui définissait sa propriété `'cursor'` sur 'inherit' (ou si aucune règle définissant la propriété `'cursor'` ne correspondait à l'élément), le curseur de la forme serait hérité de l'élément `img` ou `object` de l'élément [image map](#) , pas du parent de l' `area` élément.

### 15.5 Widget

#### 15.5.1 Apparence native

La spécification *CSS Basic User Interface* appelle les éléments qui peuvent avoir une [apparence native widgets](#) et définit s'il faut utiliser cette [apparence native](#) en fonction de la propriété `'appearance'` . Cette logique, à son tour, dépend du fait que chaque élément est classé comme [widget dévolu](#) ou [widget non dévolu](#) . Cette section définit quels éléments correspondent à ces concepts pour HTML, quelle est leur [apparence native](#) et toute particularité de leur état [dévolu ou de leur apparence primitive](#) . [\[CSSUI\]](#)

Les éléments suivants peuvent avoir une [apparence native](#) dans le cadre de la propriété CSS `'appearance'` .

- `button`
- `input`
- `meter`
- `progress`

- [select](#)
- [textarea](#)

## 15.5.2 Disposition des boutons

Lorsqu'un élément utilise [la disposition des boutons](#) , il s'agit d'un [widget dévouable](#) et son [apparence native](#) est celle d'un bouton.

La **disposition des boutons** est la suivante :

- Si l'élément est un [button](#) élément, alors la propriété ['display'](#) devrait agir comme suit :
  - Si la valeur calculée de ['display'](#) est 'inline-grid', 'grid', 'inline-flex' ou 'flex', alors se comporte comme la valeur calculée.
  - Sinon, si la valeur calculée de ['display'](#) est une valeur telle que le [type d'affichage externe](#) est 'inline', alors se comporter comme 'inline-block'.
  - Sinon, comportez-vous comme 'flow-root'.
- L'élément est censé établir un nouveau [contexte de formatage](#) pour son contenu. Le type de ce contexte de formatage est déterminé par sa valeur ['display'](#) , comme d'habitude.
- Si l'élément est [en position absolue](#) , dans le cadre du [modèle de formatage visuel CSS](#) , agissez comme si l'élément était un [élément remplacé](#) . [\[CSS\]](#)
- Si la [valeur calculée](#) de ['inline-size'](#) est 'auto', alors la [valeur utilisée](#) est la [fit-content inline size](#) .
- Pour les besoins du mot-clé 'normal' de la propriété ['align-self'](#) , agissez comme si l'élément était un élément remplacé.
- Si l'élément est un [input](#) élément, ou s'il s'agit d'un [button](#) élément et que sa valeur calculée pour ['display'](#) n'est pas 'inline-grid', 'grid', 'inline-flex' ou 'flex', alors la boîte de l'élément a un **zone de contenu du bouton anonyme** enfant avec les comportements suivants :
  - La boîte est un [conteneur de bloc au niveau du](#) bloc qui établit un nouveau [contexte de formatage de bloc](#) (c'est-à-dire que ['display'](#) est 'flow-root').
  - Si la boîte ne déborde pas sur l'axe horizontal, alors elle est centrée horizontalement.
  - Si la boîte ne déborde pas dans l'axe vertical, alors elle est centrée verticalement.

Sinon, il n'y a pas [de zone de contenu de bouton anonyme](#) .

Besoin de définir l' [apparence primitive](#) attendue .

### 15.5.3 L' [button](#) élément

L' [button](#) élément, lorsqu'il génère une [boîte CSS](#) , est censé représenter un bouton et utiliser [une disposition de bouton](#) dont le contenu de [la boîte de contenu de bouton anonyme](#) (s'il existe une [boîte de contenu de bouton anonyme](#) ) sont les boîtes enfants que la boîte de l'élément aurait autrement.

### 15.5.4 Les éléments [details](#) et [summary](#)

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
details > summary:first-of-type {
```

```
  display: list-item;
```

```
  counter-increment: list-item 0;
```

```
  list-style: disclosure-closed inside;
```

```
}
```

```
details[open] > summary:first-of-type {
```

```
  list-style-type: disclosure-open;
```

```
}
```

L' [details](#) élément est censé s'afficher sous la forme d'une [boîte de bloc](#) . L'élément devrait également avoir un [arbre fantôme](#) interne avec deux [slots](#) . Le premier [emplacement](#) est censé prendre le [details](#) premier élément enfant de l'élément [summary](#), le cas échéant. Le deuxième [emplacement](#) est censé prendre les [details](#) descendants restants de l'élément, s'il y en a.

Le [details](#) premier [summary](#) élément enfant de l'élément, le cas échéant, devrait permettre à l'utilisateur de demander que les détails soient affichés ou masqués.

Le `details` deuxième [emplacement](#) de l'élément est censé avoir son `style` attribut défini sur " `display: block; content-visibility: hidden;`" lorsque l' `details` élément n'a pas d' `open` attribut. Lorsqu'il a l' `open` attribut, l' `style` attribut est censé être supprimé du deuxième [emplacement](#) .

*Étant donné que les emplacements sont cachés à l'intérieur d'un arbre fantôme, cet `style` attribut n'est pas directement visible pour le code de l'auteur. Ses impacts sont cependant visibles. Notamment, le choix de `content-visibility: hidden` au lieu de, par exemple, `display: none` a un impact sur les résultats de diverses API qui interrogent les informations de mise en page.*

### 15.5.5 L' `input` élément en tant que widget de saisie de texte

Un `input` élément dont `type` l'attribut est à l' état [Text](#) , [Telephone](#) , [URL](#) ou [Email](#) , est un [widget dévolu](#) . Son [apparence native](#) attendue est de s'afficher sous la forme d'une boîte [de « bloc en ligne »](#) représentant un contrôle de texte sur une ligne.

Un `input` élément dont `type` l'attribut est à l' état [Recherche](#) est un [widget dévolu](#) . Son [apparence native](#) attendue est de s'afficher sous la forme d'une boîte [de « bloc en ligne »](#) représentant un contrôle de texte sur une ligne. Si la [valeur calculée](#) de la propriété '`appearance`' de l'élément n'est pas '`textfield`', il peut avoir un style distinct indiquant qu'il s'agit d'un champ de recherche.

Un `input` élément dont `type` l'attribut est à l' état [Mot de passe](#) est un [widget dévouable](#) . Son [apparence native](#) attendue est de s'afficher sous la forme d'une boîte [de « bloc en ligne »](#) représentant un contrôle de texte d'une ligne qui masque la saisie de données.

Pour `input` les éléments dont `type` l'attribut est dans l'un des états ci-dessus, la [valeur utilisée](#) de la propriété '`line-height`' doit être une valeur de longueur qui n'est pas inférieure à ce que serait la [valeur utilisée pour 'line-height: normal'](#).

*La [valeur utilisée](#) ne sera pas le mot-clé réel 'normal'. De plus, cette règle n'affecte pas la [valeur calculée](#) .*

Si ces contrôles de texte fournissent une sélection de texte, alors, lorsque l'utilisateur modifie la sélection actuelle, l'agent utilisateur est censé mettre en file d'attente [une tâche d'élément](#) sur la [source de tâche d'interaction de l'utilisateur](#) étant donné l' `input` élément pour [déclencher un événement](#) nommé `select` à l'élément, avec l' `bubbles` attribut initialisé à vrai.

Si un `input` élément dont `type` l'attribut est dans l'un des états ci-dessus a un `size` attribut, et que l'analyse de la valeur de cet attribut à l'aide des [règles d'analyse des entiers non négatifs](#) ne génère pas d'erreur, alors l'agent utilisateur est censé utiliser l'attribut comme [présentation indice](#) pour la propriété '`width`' sur

l'élément, avec la valeur obtenue en appliquant l' algorithme [de conversion d'une largeur de caractère en pixels](#) à la valeur de l'attribut.

Si un [input](#)élément dont [type](#)l'attribut est dans l'un des états ci-dessus n'a pas d' [size](#) attribut, alors l'agent utilisateur est censé agir comme s'il avait une règle de feuille de style au niveau de l'agent utilisateur définissant la propriété '[width](#)' de l'élément sur la valeur obtenue en appliquant l' algorithme [de conversion d'une largeur de caractère en pixels](#) au nombre 20.

L' algorithme **de conversion d'une largeur de caractère en pixels renvoie** (  $size - 1 \times avg + max$  , où *size* est la largeur de caractère à convertir, *avg* est la largeur de caractère moyenne de la police principale de l'élément pour lequel l'algorithme est exécuté, dans pixels, et *max* est la largeur de caractère maximale de cette même police, également en pixels. (La propriété '[letter-spacing](#)' de l'élément n'affecte pas le résultat.)

Ces contrôles de texte sont censés être [des conteneurs de défilement](#) et prendre en charge le défilement dans l' [axe en ligne](#) , mais pas dans l' [axe du bloc](#) .

Besoin de détailler l' [apparence native](#) et l'[apparence primitive](#) attendues .

### 15.5.6 L' [input](#)élément en tant que widgets spécifiques à un domaine

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Date est un widget dévouable](#) censé s'afficher sous la forme d'une boîte '[en ligne-bloc](#)' représentant un [contrôle de date](#).

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Mois](#) est un [widget dévouable](#) censé s'afficher sous la forme d'une boîte [« bloc en ligne »](#) représentant un [contrôle de mois](#).

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Semaine est un widget dévouable](#) censé s'afficher sous la forme d'une boîte [« bloc en ligne »](#) représentant un [contrôle semaine](#).

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Time est un widget dévouable](#) censé s'afficher sous la forme d'une boîte '[inline-block](#)' représentant un [contrôle de temps](#).

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Date et heure locales est un widget dévouable](#) censé s'afficher sous la forme d'une boîte de [« bloc en ligne »](#) représentant un [contrôle de date et d'heure locale](#).

Un `input` élément dont `type` l'attribut est dans l'état [Number est un widget dévolable](#) censé s'afficher sous la forme d'une boîte `'inline-block'` représentant un contrôle numérique.

Ces commandes doivent toutes avoir une hauteur d'environ une ligne et être à peu près aussi larges que nécessaire pour afficher la valeur la plus large possible.

Besoin de détailler l' [apparence native](#) et l'[apparence primitive](#) attendues .

### 15.5.7 L' `input` élément comme contrôle de distance

Un `input` élément dont `type` l'attribut est dans l'état [Range](#) est un [widget non dévolu](#) . Son [apparence native](#) attendue est de s'afficher sous la forme d'une boîte [de « bloc en ligne »](#) représentant un contrôle de curseur.

Lorsque le contrôle est plus large que haut (ou carré), le contrôle est censé être un curseur horizontal, avec la valeur la plus basse à droite si la propriété `'direction'` de cet élément a une valeur calculée [de ' rtl '](#) , et à gauche sinon. Lorsque le contrôle est plus haut que large, on s'attend à ce qu'il s'agisse d'un curseur vertical, avec la valeur la plus basse en bas.

Les valeurs suggérées prédéfinies (fournies par l' [list](#) attribut) doivent être affichées sous forme de graduations sur le curseur, auxquelles le curseur peut s'aligner.

Les agents utilisateurs sont censés utiliser la [valeur utilisée](#) de la propriété `'direction'` sur l'élément pour déterminer la direction dans laquelle le curseur fonctionne. En règle générale, un contrôle horizontal de gauche à droite ("ltr") aurait la valeur la plus basse à gauche et la valeur la plus élevée à droite, et vice versa.

Besoin de détailler l' [aspect primitif](#) attendu .

### 15.5.8 L' `input` élément comme puits de couleur

Un `input` élément dont `type` l'attribut est dans l'état [Color](#) est censé représenter un puits de couleur qui, lorsqu'il est activé, fournit à l'utilisateur un sélecteur de couleurs (par exemple une roue chromatique ou une palette de couleurs) à partir duquel la couleur peut être modifiée. L'élément, lorsqu'il génère une [boîte CSS](#) , est censé utiliser [la disposition des boutons](#) , qui n'a pas de boîtes enfants de la [boîte de contenu du bouton anonyme](#) . La [zone de contenu du bouton anonyme](#) devrait avoir un [indice de présentation](#) définissant la propriété `'background-color'` sur la [valeur](#) de l'élément .

Les valeurs suggérées prédéfinies (fournies par l' [list](#) attribut) doivent être affichées dans l'interface du sélecteur de couleurs, et non sur le puits de couleur lui-même.

Besoin de détailler l' [apparence native](#) et l'[apparence primitive](#) attendues .

#### 15.5.9 L' [input](#)élément en tant que widgets de case à cocher et de bouton radio

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Checkbox](#) est un [widget non dévolable](#) censé s'afficher sous la forme d'une boîte '[inline-block](#)' contenant un seul contrôle de case à cocher, sans étiquette.

Besoin de détailler l' [apparence native](#) et l'[apparence primitive](#) attendues .

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Bouton radio](#) est un [widget non dévolable](#) censé s'afficher sous la forme d'une boîte '[bloc en ligne](#)' contenant un seul contrôle de bouton radio, sans étiquette.

Besoin de détailler l' [apparence native](#) et l'[apparence primitive](#) attendues .

#### 15.5.10 L' [input](#)élément en tant que contrôle de téléchargement de fichier

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Téléchargement de fichier](#) , lorsqu'il génère une [boîte CSS](#) , est censé s'afficher sous la forme d'une boîte '[bloc en ligne](#)' contenant une étendue de texte donnant le(s) nom(s) de fichier des [fichiers sélectionnés](#) , le cas échéant, suivis par un bouton qui, lorsqu'il est activé, fournit à l'utilisateur un sélecteur de fichiers à partir duquel la sélection peut être modifiée. Le bouton doit utiliser [la disposition des boutons](#) et correspondre au pseudo-élément '[::file-selector-button](#)' . Le contenu de sa [zone de contenu de bouton anonyme](#) devrait être un texte [défini par l'implémentation \(et éventuellement spécifique aux paramètres régionaux\)](#), par exemple "[Choisir un fichier](#)".

#### 15.5.11 L' [input](#)élément sous forme de bouton

Un [input](#)élément dont [type](#)l'attribut est dans l' état [Submit Button](#) , [Reset Button](#) ou [Button](#) , lorsqu'il génère une [CSS box](#) , doit représenter un bouton et



utiliser [la disposition des boutons](#) et le contenu de la [zone de contenu du bouton anonyme](#) doit être le texte de l' `value` attribut de l'élément, le cas échéant, ou le texte dérivé de l'attribut de l'élément `type` d'une manière [définie par l'implémentation](#) (et probablement spécifique aux paramètres régionaux), si ce n'est pas le cas.

### 15.5.12 L' `marquee` élément

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
marquee {
```

```
display: inline-block;
```

```
text-align: initial;
```

```
}
```

L' `marquee` élément, lorsqu'il [est activé](#) , doit s'afficher de manière animée en fonction de ses attributs comme suit :

#### Si l' `behavior` attribut de l'élément est dans l' état [de défilement](#)

Faites glisser le contenu de l'élément dans la direction décrite par l' `direction` attribut tel que défini ci-dessous, de sorte qu'il commence par le côté de départ du `marquee` et se termine au ras du côté de fin interne.

Par exemple, si l' `direction` attribut est [à gauche](#) (valeur par défaut), alors le contenu commencerait de telle sorte que son bord gauche soit hors du côté du bord droit de la [zone de contenu](#) `marquee` de , et le contenu glisserait alors jusqu'au point où le le bord gauche du contenu affleure le bord intérieur gauche de la [zone de contenu](#) de `.marquee`

Une fois l'animation terminée, l'agent utilisateur doit [incrémenter l'index de la boucle courante du chapiteau](#) . Si l'élément est toujours [activé](#) après cela, l'agent utilisateur est censé redémarrer l'animation.

#### Si l' `behavior` attribut de l'élément est à l' état [de diapositive](#)

Faites glisser le contenu de l'élément dans la direction décrite par l' `direction` attribut tel que défini ci-dessous, de sorte qu'il commence par le côté début du `marquee`, et se termine par le côté fin du `marquee`.

Par exemple, si l' `direction` attribut est [à gauche](#) (valeur par défaut), alors le contenu commencerait de telle sorte que son bord gauche soit hors du côté du bord droit de la [zone de contenu](#) `marquee` de , et le contenu glisserait alors

jusqu'au point où le le bord *droit* du contenu affleure le bord intérieur gauche de la [zone de contenu](#) de [.marquee](#)

Une fois l'animation terminée, l'agent utilisateur doit [incrémenter l'index de la boucle courante du chapiteau](#) . Si l'élément est toujours [activé](#) après cela, l'agent utilisateur est censé redémarrer l'animation.

### Si l' [behavior](#) attribut de l'élément est dans l' état [alternatif](#)

Lorsque l' [index de la boucle actuelle du chapiteau](#) est pair (ou nul), faites glisser le contenu de l'élément dans la direction décrite par l' [direction](#) attribut tel que défini ci-dessous, de sorte qu'il commence au ras du côté de début de [marquee](#), et se termine au ras du côté de fin de le [marquee](#).

Lorsque l' [index de la boucle courante du chapiteau](#) est impair, faites glisser le contenu de l'élément dans la direction opposée à celle décrite par l' [direction](#) attribut tel que défini ci-dessous, de sorte qu'il commence au ras du côté de fin du [marquee](#), et se termine au ras du côté de départ du [marquee](#).

Par exemple, si l' [direction](#) attribut est [à gauche](#) (valeur par défaut), alors le contenu affleurerait avec son bord droit avec le bord intérieur droit de la [zone de contenu](#) [marquee](#) de , et le contenu glisserait alors jusqu'au point où le bord *gauche* de la le contenu affleure le bord intérieur gauche de la [zone de contenu](#) de [.marquee](#)

Une fois l'animation terminée, l'agent utilisateur doit [incrémenter l'index de la boucle courante du chapiteau](#) . Si l'élément est toujours [activé](#) après cela, l'agent utilisateur est censé continuer l'animation.

L' [direction](#) attribut a les significations décrites dans le tableau suivant :

| <a href="#">direction</a> état d'attribut | Direction de l'animation | Arête de départ | Bord d'extrémité | Direction opposée       |
|-------------------------------------------|--------------------------|-----------------|------------------|-------------------------|
| <a href="#">gauche</a>                    | ← De droite à gauche     | Droite          | Gauche           | → De gauche à droite    |
| <a href="#">droite</a>                    | → De gauche à droite     | Gauche          | Droite           | ← De droite à gauche    |
| <a href="#">en haut</a>                   | ↑ Haut (de bas en haut)  | Bas             | Haut             | ↓ Bas (de haut en bas)  |
| <a href="#">bas</a>                       | ↓ Bas (de haut en bas)   | Haut            | Bas              | ↑ Haut (de bas en haut) |

Dans tous les cas, l'animation doit se dérouler de telle sorte qu'il y ait un délai donné par l' [intervalle de défilement du chapiteau](#) entre chaque image, et de sorte que le contenu se déplace au plus de la distance donnée par la [distance de défilement du chapiteau](#) avec chaque image.

Lorsqu'un [marquee](#) élément a un [bgcolor](#) ensemble d'attributs, la valeur doit être analysée en utilisant les [règles d'analyse d'une valeur de couleur héritée](#) , et si cela ne renvoie pas d'erreur, l'agent utilisateur est censé traiter l'attribut comme un [indice](#)

de présentation définissant la valeur de l'élément propriété '[background-color](#)' à la couleur résultante.

Les attributs `width` et `height` d'un [marquee](#) élément correspondent respectivement aux propriétés de dimension '[width](#)' et '[height](#)' de l'élément.

La hauteur intrinsèque d'un [marquee](#) élément avec son [direction](#) attribut dans les états [haut](#) ou [bas](#) est de 200 [pixels CSS](#) .

L' `vspace` attribut d'un [marquee](#) élément correspond aux propriétés de dimension « [margin-top](#) » et « [margin-bottom](#) » sur l'élément. L' `hspace` attribut d'un [marquee](#) élément correspond aux propriétés de dimension '[margin-left](#)' et '[margin-right](#)' de l'élément.

La propriété '[overflow](#)' [marquee](#) de l' élément devrait être ignorée ; le débordement devrait toujours être caché.

### 15.5.13 L' [meter](#) élément

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
meter { appearance: auto; }
```

L' [meter](#) élément est un [widget dévolable](#) . [Son apparence native](#) attendue est de rendre une boîte '[inline-block](#)' avec une '[block-size](#)' de '1em' et une '[inline-size](#)' de '5em', un '[vertical-align](#)' de '-0.2em' , et dont le contenu représente une jauge.

Lorsque l'élément est plus large que haut (ou carré), la représentation devrait être d'une jauge horizontale, avec la valeur minimale à droite si la propriété 'direction' sur cet élément a une valeur calculée de '[rtl](#)' , et à gauche sinon. Lorsque l'élément est plus haut que large, on s'attend à ce qu'il représente une jauge verticale, avec la valeur minimale en bas.

Les agents utilisateurs sont censés utiliser une présentation cohérente avec les conventions de la plate-forme pour les jauges, le cas échéant.

*Les exigences relatives à ce qui doit être représenté dans la jauge sont incluses dans la définition de l' [meter](#) élément.*

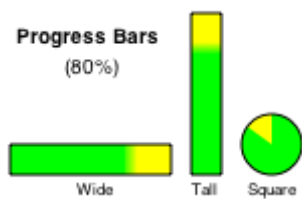
Besoin de détailler l' [aspect primitif](#) attendu .

### 15.5.14 L' progressélément

```
@namespace url(http://www.w3.org/1999/xhtml);
```

```
progress { appearance: auto; }
```

L' progressélément est un widget dévolable . Son apparence native attendue est de rendre une boîte 'inline-block' avec une 'block-size' de '1em' et une 'inline-size' de '10em', et un 'vertical-align' de '-0.2em '.



Lorsque l'élément est plus large que haut, l'élément doit être représenté sous la forme d'une barre de progression horizontale, avec le début à droite et la fin à gauche si la propriété 'direction' de cet élément a une valeur calculée de ' rtl', et avec le début à gauche et la fin à droite sinon. Lorsque l'élément est plus haut que large, on s'attend à ce qu'il soit représenté sous la forme d'une barre de progression verticale, avec la valeur la plus basse en bas. Lorsque l'élément est carré, on s'attend à ce qu'il soit représenté comme un widget de progression indépendant de la direction (par exemple, un anneau de progression circulaire).

Les agents utilisateurs doivent utiliser une présentation cohérente avec les conventions de la plate-forme pour les barres de progression. En particulier, les agents utilisateurs sont censés utiliser des présentations différentes pour les barres de progression déterminées et indéterminées. Les agents utilisateurs sont également censés modifier la présentation en fonction des dimensions de l'élément.

Par exemple, sur certaines plates-formes pour afficher la progression indéterminée, il existe un indicateur de progression "spinner" aux dimensions carrées, qui pourrait être utilisé lorsque l'élément est carré, et une barre de progression indéterminée, qui pourrait être utilisée lorsque l'élément est large.

*Les exigences concernant la manière de déterminer si la barre de progression est déterminée ou indéterminée, et la progression qu'une barre de progression déterminée doit afficher, sont incluses dans la définition de l' progressélément.*

Besoin de détailler l' aspect primitif attendu .

### 15.5.15 L' selectélément

Un `select` élément est soit une **liste déroulante** , soit une **liste déroulante** , selon ses attributs.

Un `select` élément dont `multiple` l'attribut est présent doit s'afficher sous la forme d'une zone de liste à sélection multiple .

Un `select` élément dont `multiple` l'attribut est absent et dont la taille d'affichage est supérieure à 1 doit s'afficher sous la forme d'une zone de liste à sélection unique .

Lorsque l'élément s'affiche sous la forme d'une zone de liste , il s'agit d'un widget dévolable censé s'afficher sous la forme d'une boîte `'inline-block'` dont la `'hauteur'` est la hauteur nécessaire pour contenir autant de lignes pour les éléments que donné par la taille d'affichage de l'élément , ou quatre rows si l'attribut est absent, et dont la `'width'` est la largeur des `select` étiquettes de ' plus la largeur d'une barre de défilement.

Un `select` élément dont `multiple` l'attribut est absent, et dont la taille d'affichage est 1, est censé s'afficher sous la forme d'une liste déroulante d'une ligne `'inline-block'` dont la largeur est la largeur des étiquettes de `.select`

Lorsque l'élément s'affiche sous la forme d'une liste déroulante , il s'agit d'un widget dévolable . Son apparence dans l'état dévolu, ainsi que son apparence lorsque la valeur calculée de la propriété `'appearance'` de l'élément est `'menulist-button'`, est celle d'une liste déroulante, incluant un "bouton déroulant", mais pas nécessairement rendu à l'aide d'un natif contrôle du système d'exploitation hôte. Dans un tel état, les propriétés CSS telles que `'color'` , `'background-color'` et `'border'` ne doivent pas être ignorées (comme c'est généralement permis lors du rendu d'un élément en fonction de son apparence native ).

Dans les deux cas ( liste déroulante ou liste déroulante ), les éléments de l'élément sont censés être la liste d'options de l'élément , les `optgroup` éléments enfants de l'élément fournissant des en-têtes pour les groupes d'options, le cas échéant.

Un `optgroup` élément doit être rendu en affichant l' label attribut de l'élément.

Un `option` élément est censé être rendu en affichant le label de l'élément , en retrait sous son `optgroup` élément s'il en a un.

La **largeur des `select` étiquettes de** est la plus large entre la largeur nécessaire pour rendre le plus large `optgroup` et la largeur nécessaire pour rendre l' `option` élément le plus large dans la liste d'options de l'élément (y compris son retrait, le cas échéant).

Si un `select` élément contient un espace réservé label option , l'agent utilisateur est censé le restituer `option` d'une manière qui indique qu'il s'agit d'un label plutôt que d'une option valide du contrôle. Cela peut inclure le fait d'empêcher l' option d'étiquette d'espace réservé d'être explicitement sélectionnée par l'utilisateur. Lorsque la sélection de l' option d'étiquette d'espace réservé est vraie, le

contrôle doit s'afficher de manière à indiquer qu'aucune option valide n'est actuellement sélectionnée.

Les agents utilisateurs sont censés restituer les étiquettes de [select](#) manière à ce que tout alignement reste cohérent, que l'étiquette soit affichée dans le cadre de la page ou dans un contrôle de menu.

Besoin de détailler l' [apparence native](#) et l'[apparence primitive](#) attendues .

### 15.5.16 L' [textarea](#)élément

L' [textarea](#)élément est un [widget dévouable](#) censé s'afficher sous la forme d'une boîte « [bloc en ligne](#) » représentant un contrôle de texte multiligne. Si ce contrôle de texte multiligne fournit une sélection, alors, lorsque l'utilisateur modifie la sélection actuelle, l'agent utilisateur est censé mettre en file d'attente [une tâche d'élément](#) sur la [source de la tâche d'interaction utilisateur](#) étant donné l' [textarea](#)élément pour [déclencher un événement](#) nommé [select](#) à l'élément, avec l' [bubbles](#)attribut initialisé à vrai.

Si l'élément a un [cols](#)attribut et que l'analyse de la valeur de cet attribut à l'aide des [règles d'analyse des entiers non négatifs](#) ne génère pas d'erreur, l'agent utilisateur est censé utiliser l'attribut comme [indice de présentation](#) pour la propriété '[width](#)' sur le élément, la valeur étant la [largeur effective de la zone de texte](#) (telle que définie ci-dessous). Sinon, l'agent utilisateur est censé agir comme s'il avait une règle de feuille de style au niveau de l'agent utilisateur définissant la propriété '[width](#)' de l'élément sur la [textarea effective width](#) .

La **largeur effective de la zone de texte** d'un [textarea](#)élément est  $size \times avg + sbw$  , où *size* est la [largeur de caractère](#) de l'élément , *avg* est la largeur de caractère moyenne de la police principale de l'élément, en [pixels CSS](#) , et *sbw* est la largeur d'une barre de défilement, en [pixels CSS](#) . (La propriété '[letter-spacing](#)' de l'élément n'affecte pas le résultat.)

Si l'élément a un [rows](#)attribut et que l'analyse de la valeur de cet attribut à l'aide des [règles d'analyse des entiers non négatifs](#) ne génère pas d'erreur, l'agent utilisateur est censé utiliser l'attribut comme [indice de présentation](#) pour la propriété '[height](#)' sur le élément, la valeur étant la [hauteur effective de la zone de texte](#) (telle que définie ci-dessous). Sinon, l'agent utilisateur est censé agir comme s'il avait une règle de feuille de style au niveau de l'agent utilisateur définissant la propriété '[height](#)' de l'élément sur la [textarea effective height](#) .

La **hauteur effective de la zone de texte** d'un [textarea](#)élément est la hauteur en [pixels CSS](#) du nombre de lignes spécifiées dans la [hauteur de caractère](#) de l'élément , plus la hauteur d'une barre de défilement en [pixels CSS](#) .

Les agents utilisateurs doivent appliquer la propriété CSS '[white-space](#)' aux `textarea` éléments. Pour des raisons historiques, si l'élément a un `wrap` attribut dont la valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " `off`", alors l'agent utilisateur est censé traiter l'attribut comme un [indice de présentation](#) en définissant la propriété '[white-space](#)' de l'élément sur ' `pré`'.

Besoin de détailler l' [apparence native](#) et l'[apparence primitive](#) attendues .

## 15.6 Cadres et jeux de cadres

L'agent utilisateur doit restituer `frameset` les éléments sous la forme d'une boîte avec la hauteur et la largeur de la [fenêtre](#) , avec une surface rendue selon l'algorithme de disposition suivant :

1. Les variables *cols* et *rows* sont des listes de zéro ou plusieurs paires composées d'un nombre et d'une unité, l'unité étant l'une des *pourcentage* , *relatif* et *absolu* .

Utilisez les [règles d'analyse d'une liste de dimensions](#) pour analyser la valeur de l' `cols` attribut de l'élément, s'il y en a un. Soit *cols* le résultat, ou une liste vide s'il n'y a pas un tel attribut.

Utilisez les [règles d'analyse d'une liste de dimensions](#) pour analyser la valeur de l' `rows` attribut de l'élément, s'il y en a un. Soit *les lignes* comme résultat, ou une liste vide s'il n'y a pas un tel attribut.

2. Pour toutes les entrées dans *les colonnes* ou *les lignes* qui ont le numéro zéro et l'unité *relative* , remplacez le numéro de l'entrée par un.
3. Si *cols* n'a pas d'entrées, ajoutez une seule entrée composée de la valeur 1 et de l'unité *relative* à *cols* .

Si *rows* n'a pas d'entrées, ajoutez une seule entrée composée de la valeur 1 et de l'unité *relative* à *rows* .

4. Appelez l'algorithme défini ci-dessous pour [convertir une liste de dimensions en une liste de valeurs de pixels](#) en utilisant *cols* comme liste d'entrée et la largeur de la surface dans laquelle `frameset` est rendu, en [pixels CSS](#) , comme dimension d'entrée. Soit *cols de taille* la liste résultante.

Appelez l'algorithme défini ci-dessous pour [convertir une liste de dimensions en une liste de valeurs de pixels](#) en utilisant *des lignes* comme liste d'entrée et la hauteur de la surface dans laquelle `frameset` est rendu, en [pixels CSS](#) , comme dimension d'entrée. Soit *les lignes dimensionnées* comme la liste résultante.



5. Divisez la surface en une grille de rectangles  $w \times h$ , où  $w$  est le nombre d'entrées dans *les colonnes dimensionnées* et  $h$  est le nombre d'entrées dans *les lignes dimensionnées*.

Dimensionnez les colonnes de sorte que chaque colonne de la grille ait autant [de pixels CSS](#) de large que l'entrée correspondante dans la liste *des colonnes dimensionnées*.

Dimensionnez les lignes de sorte que chaque ligne de la grille ait autant [de pixels CSS](#) de haut que l'entrée correspondante dans la liste *des lignes dimensionnées*.

6. Soit *children* la liste des éléments [frame](#) et [frameset](#) qui sont [des enfants](#) de l' [frameset](#) élément pour lequel l'algorithme a été invoqué.
7. Pour chaque ligne de la grille de rectangles créée à l'étape précédente, de haut en bas, exécutez ces sous-étapes :
  1. Pour chaque rectangle de la ligne, de gauche à droite, exécutez ces sous-étapes :
    1. S'il reste des éléments dans *children*, prenez le premier élément de la liste et affectez-le au rectangle.

S'il s'agit d'un [frameset](#) élément, récursez l'intégralité de [frameset](#) l'algorithme de mise en page pour cet [frameset](#) élément, avec le rectangle comme surface.

Sinon, c'est un [frame](#) élément; rendre son [contenu navigable](#), positionné et dimensionné pour s'adapter au rectangle.
    2. S'il reste des éléments dans *children*, supprimez le premier élément de *children*.
8. Si l' [frameset](#) élément [a une bordure](#), dessinez un ensemble extérieur de bordures autour des rectangles, en utilisant la [couleur de bordure de cadre](#) de l'élément.

Pour chaque rectangle, s'il y a un élément assigné à ce rectangle, et que cet élément [a une bordure](#), dessinez un ensemble intérieur de bordures autour de ce rectangle, en utilisant la [couleur de bordure de cadre](#) de l'élément.

Pour chaque bordure (visible) qui ne jouxte pas un rectangle auquel est assigné un [frame](#) élément avec un `noresize` attribut (y compris les rectangles dans d'autres [frameset](#) éléments imbriqués), l'agent utilisateur est censé permettre à l'utilisateur de déplacer la bordure, de redimensionner les rectangles à l'intérieur, de conserver les proportions de toutes [frameset](#) les grilles imbriquées.

Un élément [frameset](#) **or a une bordure** si l'algorithme suivant renvoie true : [frame](#)



1. Si l'élément a un `frameborder`attribut dont la valeur n'est pas la chaîne vide et dont le premier caractère est soit un caractère U+0031 CHIFFRE UN (1), soit un caractère U+0079 LETTRE MINUSCULE LATINE Y (y), soit un caractère U+0059 MAJUSCULE LATINE caractère LETTRE Y (Y), puis renvoie vrai.
2. Sinon, si l'élément a un `frameborder`attribut, retourne false.
3. Sinon, si l'élément a un élément parent qui est un `frameset`élément, alors retournez true si cet élément [a une bordure](#) et false si ce n'est pas le cas.
4. Sinon, retourne vrai.

La **couleur de la bordure du cadre** d'un élément `frameset`ou `frame`est la couleur obtenue à partir de l'algorithme suivant :

5. Si l'élément a un `bordercolor`attribut et que l'application des [règles d'analyse d'une valeur de couleur héritée](#) à la valeur de cet attribut ne génère pas d'erreur, renvoie la couleur ainsi obtenue.
6. Sinon, si l'élément a un élément parent qui est un `frameset`élément, renvoie la [couleur de la bordure du cadre](#) de cet élément.
7. Sinon, retourne gris.

L'algorithme de **conversion d'une liste de dimensions en une liste de valeurs de pixels** comprend les étapes suivantes :

1. Soit *la liste d'entrée* la liste des nombres et des unités transmises à l'algorithme.

Soit *output list* une liste de nombres de la même longueur que *input list* , tous nuls.

Les entrées de *la liste de sortie* correspondent aux entrées de *la liste d'entrée* qui ont la même position.

2. Soit *la dimension d'entrée* la taille transmise à l'algorithme.
3. Soit *count percent* le nombre d'entrées dans *la liste d'entrée* dont l'unité est *le pourcentage* .

Soit *le pourcentage total* soit la somme de tous les nombres de *la liste d'entrée* dont l'unité est *le pourcentage* .

Soit *count relative* le nombre d'entrées dans *la liste d'entrée* dont l'unité est *relative* .

Soit *total relatif* la somme de tous les nombres de *la liste d'entrée* dont l'unité est *relative* .

Soit *count absolu* le nombre d'entrées dans *la liste d'entrée* dont l'unité est *absolue* .

Soit *total absolue* la somme de tous les nombres de *la liste d'entrée* dont l'unité est *absolue* .

Soit *l'espace restant* la valeur de *la dimension d'entrée* .

4. Si *le total absolu* est supérieur à *l'espace restant* , alors pour chaque entrée dans *la liste d'entrée* dont l'unité est *absolue* , définissez la valeur correspondante dans *la liste de sortie* au numéro de l'entrée dans *la liste d'entrée* multiplié par *l'espace restant* et divisé par *le total absolu* . Ensuite, réglez *l'espace restant* sur zéro.

Sinon, pour chaque entrée dans *la liste d'entrée* dont l'unité est *absolue* , définissez la valeur correspondante dans *la liste de sortie* au numéro de l'entrée dans *la liste d'entrée* . Ensuite, décrémente *l'espace restant* de *total absolue* .

5. Si *le pourcentage total* multiplié par la *dimension d'entrée* et divisé par 100 est supérieur à *l'espace restant* , alors pour chaque entrée dans *la liste d'entrée* dont l'unité est *le pourcentage* , définissez la valeur correspondante dans *la liste de sortie* au numéro de l'entrée dans *la liste d'entrée* multiplié par *l'espace restant* et divisé par *le pourcentage total* . Ensuite, réglez *l'espace restant* sur zéro.

Sinon, pour chaque entrée dans *la liste d'entrée* dont l'unité est *le pourcentage* , définissez la valeur correspondante dans *la liste de sortie* sur le numéro de l'entrée dans *la liste d'entrée* multiplié par la *dimension d'entrée* et divisé par 100. Ensuite, décrémente *l'espace restant* du *pourcentage total* multiplié par la *dimension d'entrée* et divisé par 100.

6. Pour chaque entrée dans *la liste d'entrée* dont l'unité est *relative* , définissez la valeur correspondante dans *la liste de sortie* au numéro de l'entrée dans *la liste d'entrée* multiplié par *l'espace restant* et divisé par *le total relatif* .

7. Retourne *la liste de sortie* .

Les agents utilisateurs travaillant avec des valeurs entières pour les largeurs de trame (par opposition aux agents utilisateurs qui peuvent disposer les trames avec une précision sous-pixel) sont censés distribuer le reste d'abord à la dernière entrée dont l'unité est *relative* , puis de manière égale (et non proportionnelle) à chaque *entrée* dont l'unité est *pourcentage* , puis également (et non proportionnellement) à chaque entrée dont l'unité est *absolue* , et enfin, à défaut de tout le reste, à la dernière entrée.

---

Le contenu d'un `frame` élément qui n'a pas de `frameset` parent doit être rendu en [noir transparent](#) ; l'agent utilisateur ne devrait pas rendre son [contenu navigable](#) dans ce cas, et son [contenu navigable](#) devrait avoir une [fenêtre d'affichage](#) avec une largeur et une hauteur nulles.

## 15.7 Médias interactifs

### 15.7.1 Liens, formulaires et navigation

Les agents utilisateurs sont censés permettre à l'utilisateur de contrôler les aspects de l'activation [des hyperliens](#) et [de la soumission des formulaires](#) , tels que le [navigable](#) à utiliser pour la [navigation](#) ultérieure .

Les agents utilisateurs sont censés permettre aux utilisateurs de découvrir la destination des [hyperliens](#) et des [formulaires](#) avant de déclencher leur [navigation](#) .

Les agents utilisateurs sont censés informer l'utilisateur si un [lien hypertexte](#) inclut un [audit d'hyperlien](#) et lui faire savoir au minimum quels domaines seront contactés dans le cadre d'un tel audit.

Les agents utilisateurs peuvent permettre aux utilisateurs de [naviguer vers](#) les URL [indiquées](#) par les attributs sur les éléments , , et `blockquote` `insdel`

Les agents utilisateurs peuvent afficher [des hyperliens](#) créés par `link` des éléments dans leur interface utilisateur, comme discuté [précédemment](#) .

### 15.7.2 L' `title` attribut

Les agents utilisateurs sont censés exposer les [informations consultatives](#) des éléments à la demande de l'utilisateur et informer l'utilisateur de la présence de telles informations.

Sur les systèmes graphiques interactifs où l'utilisateur peut utiliser un dispositif de pointage, cela peut prendre la forme d'une info-bulle. Lorsque l'utilisateur est incapable d'utiliser un dispositif de pointage, l'agent utilisateur est censé rendre le contenu disponible d'une autre manière, par exemple en faisant de l'élément une zone focalisable [et](#) en affichant toujours les [informations de conseil](#) de l'élément actuellement [focalisé](#) , ou en affichant les [informations de conseil](#) des éléments sous le doigt de l'utilisateur sur un dispositif tactile lorsque l'utilisateur effectue un panoramique sur l'écran.

Les caractères U+000A LINE FEED (LF) sont censés provoquer des sauts de ligne dans l'info-bulle ; Les caractères U+0009 CHARACTER TABULATION (tabulation)

doivent s'afficher sous la forme d'un décalage horizontal différent de zéro qui aligne le glyphe suivant avec le taquet de tabulation suivant, les taquets de tabulation se produisant à des points qui sont des multiples de 8 fois la largeur d'un ESPACE U+0020 personnage.

Par exemple, un agent utilisateur visuel pourrait rendre les éléments avec un `title`attribut [focusable](#) et pourrait faire en sorte que tout élément [focalisé](#) avec un `title`attribut affiche son info-bulle sous l'élément pendant que l'élément a le focus. Cela permettrait à un utilisateur de tabuler dans le document pour trouver tout le texte consultatif.

Comme autre exemple, un lecteur d'écran pourrait fournir un signal audio lors de la lecture d'un élément avec une info-bulle, avec une touche associée pour lire la dernière info-bulle pour laquelle un signal a été joué.

### 15.7.3 Modification des hôtes

Le curseur d'édition de texte actuel (c'est-à-dire la [plage active](#) , s'il est vide et dans un [hôte d'édition](#) ), le cas échéant, devrait agir comme un [élément remplacé](#) en ligne avec les dimensions verticales du curseur et avec une largeur nulle pour les besoins du Modèle de rendu CSS.

*Cela signifie que même un bloc vide peut avoir le caret à l'intérieur, et que lorsque le caret est dans un tel élément, il empêche [les marges de s'effondrer](#) à travers l'élément.*

### 15.7.4 Texte rendu dans les interfaces utilisateur natives

Les agents utilisateurs sont censés respecter la sémantique Unicode du texte exposé dans les interfaces utilisateur, par exemple en prenant en charge l'algorithme bidirectionnel dans le texte affiché dans les boîtes de dialogue, les barres de titre, les menus contextuels et les info-bulles. Le texte du contenu des éléments doit être rendu d'une manière qui respecte [la directionnalité](#) de l'élément à partir duquel le texte a été obtenu. Le texte des attributs doit être restitué d'une manière qui respecte [la directionnalité de l'attribut](#) .

Considérez le balisage suivant, qui contient du texte hébreu demandant un langage de programmation, les langages étant du texte pour lequel une direction de gauche à droite est importante compte tenu de la ponctuation dans certains de leurs noms :

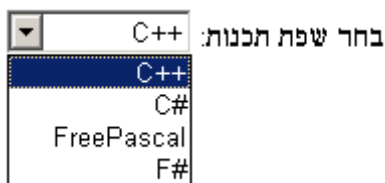
```
<p dir="rtl" lang="he">
  <label>
    בחר שפת תכנות:
```

```

<select>
  <option dir="ltr">C++</option>
  <option dir="ltr">C#</option>
  <option dir="ltr">FreePascal</option>
  <option dir="ltr">F#</option>
</select>
</label>
</p>

```

Si l' select élément était rendu sous la forme d'une liste déroulante, un rendu correct garantirait que la ponctuation était la même à la fois dans la liste déroulante et dans la zone affichant la sélection actuelle.



La directionnalité des attributs dépend de l'attribut et de l' dir attribut de l'élément, comme le montre l'exemple suivant. Considérez ce balisage :

```

<table>
  <tr>
    <th abbr="(ⵏ" dir="ltr">A
    <th abbr="(ⵏ" dir="rtl">A
    <th abbr="(ⵏ" dir="auto">A
  </tr>
</table>

```

Si les abbr attributs sont rendus, par exemple dans une info-bulle ou une autre interface utilisateur, le premier aura une parenthèse gauche (parce que la direction est 'ltr'), le second aura une parenthèse droite (parce que la direction est 'rtl'), et le troisième aura une parenthèse fermante (parce que la direction est déterminée à partir de la valeur de l'attribut comme étant 'rtl').

Cependant, si à la place l'attribut n'était pas un attribut capable de directionnalité , les résultats seraient différents :

```

<table>
  <tr>
    <th data-abbrev="(ⵏ" dir="ltr">A
    <th data-abbrev="(ⵏ" dir="rtl">A
    <th data-abbrev="(ⵏ" dir="auto">A
  </tr>
</table>

```

Dans ce cas, si l'agent utilisateur devait exposer l' `data-abbrev` attribut dans l'interface utilisateur (par exemple dans un environnement de débogage), le dernier cas serait rendu avec une parenthèse *gauche* , car la direction serait déterminée à partir du contenu de l'élément.

Une chaîne fournie par un script (par exemple, l'argument de `window.alert()`) est censée être traitée comme un ensemble indépendant d'un ou plusieurs paragraphes d'algorithme bidirectionnel lorsqu'elle est affichée, comme défini par l'algorithme bidirectionnel, y compris, par exemple, la prise en charge du comportement de rupture de paragraphe de Caractères U+000A LINE FEED (LF). Aux fins de déterminer le niveau de paragraphe d'un tel texte dans l'algorithme bidirectionnel, la présente spécification ne prévoit *pas* de remplacement de niveau supérieur des règles P2 et P3. [\[BIDI\]](#)

Si nécessaire, les auteurs peuvent appliquer une direction particulière pour un paragraphe donné en le commençant par les caractères Unicode U+200E MARQUE DE GAUCHE À DROITE ou U+200F MARQUE DE DROITE À GAUCHE.

Ainsi, le script suivant :

```
alert('\u05DC\u05DE\u05D3 HTML \u05D4\u05D9\u05D5\u05DD!')
```

... entraînerait toujours un message indiquant " !היום LMTH למד " (et non " HTML דמל !"), quelle que soit la langue de l'interface de l'agent utilisateur ou la direction de la page ou de l'un de ses éléments.

Pour un exemple plus complexe, considérez le script suivant :

```
/* Warning: this script does not handle right-to-left scripts
correctly */
var s;
if (s = prompt('What is your name?')) {
    alert(s + '! Ok, Fred, ' + s + ', and Wilma will get the car.');
```

Lorsque l'utilisateur entre " Kitty", l'agent utilisateur alerte "Minou! Ok, Fred, Kitty et Wilma prendront la voiture.". Cependant, si l'utilisateur saisit " لا أفهم", l'algorithme bidirectionnel déterminera que la direction du paragraphe est de droite à gauche, et la sortie sera donc le désordre imprévu suivant : " لا أفهم ! derF ,kO, لا أفهم , rac eht teg lliw amliW adn."

Pour forcer une alerte commençant par un texte fourni par l'utilisateur (ou un autre texte de directionnalité inconnue) à s'afficher de gauche à droite, la chaîne peut être précédée du caractère U+200E LEFT-TO-RIGHT MARK :

```
var s;
if (s = prompt('What is your name?')) {
    alert('\u200E' + s + '! Ok, Fred, ' + s + ', and Wilma will get
the car.');
```

```
}
```

## 15.8 Supports d'impression

Les agents utilisateurs sont censés permettre à l'utilisateur de demander la possibilité d' **obtenir une forme physique** (ou une représentation d'une forme physique) d'un fichier [Document](#). Par exemple, sélectionner l'option d'imprimer une page ou de la convertir au format PDF. [\[PDF\]](#)

Lorsque l'utilisateur [obtient réellement une forme physique](#) (ou une représentation d'une forme physique) d'un [Document](#), l'agent utilisateur est censé créer un nouveau rendu du [Document](#) pour le support d'impression.

## 15.9 Documents XML sans style

Les agents utilisateurs HTML peuvent, dans certaines circonstances, se retrouver à rendre des documents non HTML qui utilisent des vocabulaires pour lesquels ils n'ont aucune connaissance intégrée. Cette section fournit un moyen pour les agents utilisateurs de gérer de tels documents d'une manière quelque peu utile.

Alors que a [Document](#) est un [document sans style](#) , l' agent utilisateur est censé rendre [une vue de document sans style](#) .

A [Document](#) est un **document sans style** tant qu'il remplit les conditions suivantes :

- Le [Document](#) n'a pas de feuilles de style d'auteur (qu'elles soient référencées par des en-têtes HTTP, des instructions de traitement, des éléments tels que [link](#), des éléments en ligne tels que [style](#), ou tout autre mécanisme).
- Aucun des éléments du [Document](#) n'a de [conseils de présentation](#) .
- Aucun des éléments du [Document](#) n'a d' [attributs de style](#) .
- Aucun des éléments du [Document](#) n'est dans l'un des espaces de noms suivants : [espace de noms HTML](#) , [espace de noms SVG](#) , [espace de noms MathML](#)
- Le [Document](#) n'a pas [de zone focalisable](#) (par exemple à partir de XLink) autre que la [fenêtre](#) .
- Le [Document](#) n'a pas [d'hyperliens](#) (par exemple depuis XLink).
- Il n'existe aucun [script](#) dont l'[objet global](#) de [l'objet paramètres](#) est un objet [associé](#) à this . [WindowDocumentDocument](#)

- Aucun des éléments du [Document](#) n'a d'écouteurs d'événements enregistrés.

**Une vue de document sans style** est celle où le DOM n'est pas rendu selon CSS (ce qui, puisqu'il n'y a pas de styles applicables dans ce contexte, résulterait simplement en un mur de texte), mais est plutôt rendu d'une manière qui est utile pour un développeur. Cela peut consister à afficher simplement la [Document](#) source de l'objet, peut-être avec une coloration syntaxique, ou à afficher uniquement l'arborescence DOM, ou simplement un message indiquant que la page n'est pas un document stylé.

*Si a [Document](#) cesse d'être un [document sans style](#) , alors les conditions ci-dessus cessent de s'appliquer, et donc un agent utilisateur qui respecte ces exigences passera à l'utilisation du rendu CSS normal.*

## 1. [16 fonctionnalités obsolètes](#)

### 1. [16.1 Caractéristiques obsolètes mais conformes](#)

#### 1. [16.1.1 Avertissements pour les fonctionnalités obsolètes mais conformes](#)

### 2. [16.2 Éléments non conformes](#)

### 3. [16.3 Exigences pour les implémentations](#)

#### 1. [16.3.1 L'\[marquee\]\(#\)élément](#)

#### 2. [16.3.2 Cadres](#)

#### 3. [16.3.3 Autres éléments, attributs et API](#)

## 16 fonctionnalités obsolètes

### 16.1 Caractéristiques obsolètes mais conformes

Les fonctionnalités répertoriées dans cette section déclencheront des avertissements dans les vérificateurs de conformité.

Les auteurs ne doivent pas spécifier d' [border](#) attribut sur un [img](#) élément. Si l'attribut est présent, sa valeur doit être la chaîne " 0". CSS devrait être utilisé à la place.

Les auteurs ne doivent pas spécifier d' [charset](#) attribut sur un [script](#) élément. Si l'attribut est présent, sa valeur doit être une correspondance [ASCII non sensible à la casse](#) pour " `utf-8`". (Cela n'a aucun effet dans un document qui se conforme aux exigences ailleurs dans cette norme d'être encodé en [UTF-8](#) .)



Les auteurs ne doivent pas spécifier d' [language](#)attribut sur un [script](#)élément. Si l'attribut est présent, sa valeur doit être une correspondance [ASCII insensible à la casse](#) pour la chaîne " JavaScript" et soit l' [type](#)attribut doit être omis, soit sa valeur doit être une correspondance [ASCII insensible à la casse](#) pour la chaîne " text/javascript". L'attribut doit être entièrement omis à la place (avec la valeur " JavaScript", il n'a aucun effet), ou remplacé par l'utilisation de l' [type](#)attribut.

Les auteurs ne doivent pas spécifier une valeur pour l' [type](#) attribut sur [script](#)les éléments qui soit la chaîne vide ou une [correspondance d'essence de type MIME JavaScript](#) . Au lieu de cela, ils doivent omettre l'attribut, qui a le même effet.

Les auteurs ne doivent pas spécifier d' [type](#)attribut sur un [style](#)élément. Si l'attribut est présent, sa valeur doit être une correspondance [ASCII non sensible à la casse](#) pour " [text/css](#)".

Les auteurs ne doivent pas spécifier l' [name](#)attribut sur [a](#)les éléments. Si l'attribut est présent, sa valeur ne doit pas être la chaîne vide et ne doit ni être égale à la valeur de l'un des ID [dans](#) l' [arborescence](#) de l'élément autre que le propre [ID](#) de l'élément , le cas échéant, ni être égale à la valeur de l'un des les autres [name](#) attributs sur [a](#)les éléments dans l' [arborescence](#) de l'élément . Si cet attribut est présent et que l'élément a un [ID](#) , alors la valeur de l'attribut doit être égale à l' [ID](#) de l'élément . Dans les versions antérieures du langage, cet attribut était conçu comme un moyen de spécifier des cibles possibles pour [les fragments](#) dans [les URL](#). L' [id](#)attribut doit être utilisé à la place.

Les auteurs ne doivent pas, mais peuvent malgré les exigences contraires ailleurs dans cette spécification, spécifier les attributs [maxlength](#)et [size](#)sur [input](#)les éléments dont [type](#)les attributs sont dans l' état [Nombre](#) . Une raison valable d'utiliser ces attributs malgré tout est d'aider les agents utilisateurs hérités qui ne prennent pas en charge [input](#)les éléments à `type="number"` rendre le contrôle de texte avec une largeur utile.

### 16.1.1 Avertissements pour les fonctionnalités obsolètes mais conformes

Pour faciliter la transition des documents HTML4 Transitional vers le langage défini dans *cette* spécification, et pour décourager certaines fonctionnalités qui ne sont autorisées que dans de très rares circonstances, les vérificateurs de conformité doivent avertir l'utilisateur lorsque les fonctionnalités suivantes sont utilisées dans un document. Il s'agit généralement d'anciennes fonctionnalités obsolètes qui n'ont aucun effet et qui ne sont autorisées qu'à faire la distinction entre les erreurs probables (erreurs de conformité régulières) et un simple balisage résiduel ou des pratiques inhabituelles et déconseillées (ces avertissements).

Les fonctionnalités suivantes doivent être classées comme décrit ci-dessus :

- La présence d'un `border` attribut sur un `img` élément si sa valeur est la chaîne " 0".
- La présence d'un `charset` attribut sur un `script` élément si sa valeur est une correspondance [ASCII insensible à la casse](#) pour " utf-8".
- La présence d'un `language` attribut sur un `script` élément si sa valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " JavaScript" et s'il n'y a pas `type` d'attribut ou s'il y en a et que sa valeur est une correspondance [ASCII insensible à la casse](#) pour la chaîne " text/javascript".
- La présence d'un `type` attribut sur un `script` élément si sa valeur est une [essence de type JavaScript MIME match](#) .
- La présence d'un `type` attribut sur un `style` élément si sa valeur est une correspondance [ASCII insensible à la casse](#) pour " text/css".
- La présence d'un `name` attribut sur un `a` élément, si sa valeur n'est pas la chaîne vide.
- La présence d'un `maxlength` attribut sur un `input` élément dont `type` l'attribut est à l' état [Nombre](#) .
- La présence d'un `size` attribut sur un `input` élément dont `type` l'attribut est à l' état [Nombre](#) .

Les vérificateurs de conformité doivent faire la distinction entre les pages qui n'ont pas d'erreurs de conformité et n'ont aucune de ces fonctionnalités obsolètes, et les pages qui n'ont pas d'erreurs de conformité mais qui ont certaines de ces fonctionnalités obsolètes.

Par exemple, un validateur pourrait signaler certaines pages comme "HTML valide" et d'autres comme "HTML valide avec avertissements".

## 16.2 Éléments non conformes

Les éléments de la liste suivante sont entièrement obsolètes et ne doivent pas être utilisés par les auteurs :

### `applet`

Utilisez `embed` ou `object` à la place.

### `acronym`

Utilisez `abbr` à la place.

### `bgsound`

Utilisez [audio](#) à la place.

#### **dir**

Utilisez [ul](#) à la place.

#### **frame**

#### **frameset**

#### **noframes**

Utilisez [iframe](#) et CSS à la place, ou utilisez les inclusions côté serveur pour générer des pages complètes avec les différentes parties invariantes fusionnées.

#### **isindex**

Utilisez plutôt une combinaison de contrôle explicite [form](#) et [de texte](#).

#### **keygen**

Pour les cas d'utilisation de la gestion des appareils en entreprise, utilisez les fonctionnalités de gestion natives sur l'appareil.

Pour les cas d'utilisation d'inscription de certificat, utilisez l'API Web Cryptography pour générer une paire de clés pour le certificat, puis exportez le certificat et la clé pour permettre à l'utilisateur de les installer manuellement. [\[WEBCRYPTO\]](#)

#### **listing**

Utilisez [pre](#) et [code](#) à la place.

#### **menuitem**

Pour implémenter un menu contextuel personnalisé, utilisez un script pour gérer l'[contextmenu](#) événement.

#### **nextid**

Utilisez plutôt des GUID.

#### **noembed**

À utiliser [object](#) à la place de [embed](#) lorsqu'un repli est nécessaire.

#### **param**

Utilisez l'[data](#) attribut de l'[object](#) élément pour définir l'URL de la ressource externe.

#### **plaintext**

Utilisez plutôt le [type MIME](#) [text/plain](#) " " .

#### **rb**

#### **rtc**

Il suffit de fournir la base ruby directement à l'intérieur de l'[ruby](#) élément ou d'utiliser des éléments imbriqués [.ruby](#)

### **strike**

Utilisez del à la place si l'élément marque une modification, sinon utilisez s à la place.

### **xmp**

Utilisez pre et code à la place, et échappez les caractères "<" et "&" comme "&lt;" et "&amp;" respectivement.

### **basefont**

### **big**

### **blink**

### **center**

### **font**

### **marquee**

### **multicol**

### **noabr**

### **spacer**

### **tt**

Utilisez plutôt les éléments appropriés ou CSS.

Là où l' tt élément aurait été utilisé pour marquer la saisie au clavier, considérez l' kbd élément ; pour les variables, considérez l' var élément ; pour le code informatique, considérez l' code élément ; et pour la sortie de l'ordinateur, considérez l' samp élément.

De même, si l' big élément est utilisé pour désigner un titre, envisagez d'utiliser l' h1 élément ; s'il est utilisé pour marquer des passages importants, considérez l' strong élément ; et s'il est utilisé pour mettre en évidence du texte à des fins de référence, considérez l' mark élément.

Voir aussi le [résumé de l'utilisation de la sémantique au niveau du texte](#) pour plus de suggestions avec des exemples.

---

Les attributs suivants sont obsolètes (bien que les éléments fassent toujours partie du langage) et ne doivent pas être utilisés par les auteurs :

### **charset** sur a les éléments

### **charset** sur link les éléments

Utilisez plutôt un en-tête HTTP ` Content-Type ` sur la ressource liée.

### **charset** sur script les éléments (sauf comme indiqué dans la section précédente)

Omettez l'attribut. Les documents et les scripts sont requis pour utiliser [UTF-8](#), il est donc redondant de le spécifier sur l' [script](#) élément puisqu'il hérite du document.

**coords** sur [a](#) les éléments

**shape** sur [a](#) les éléments

Utilisez [area](#) à la place de [a](#) pour les images cliquables.

**methods** sur [a](#) les éléments

**methods** sur [link](#) les éléments

Utilisez plutôt la fonctionnalité HTTP OPTIONS.

**name** sur [a](#) les éléments (sauf comme indiqué dans la section précédente)

**name** sur [embed](#) les éléments

**name** sur [img](#) les éléments

**name** sur [option](#) les éléments

Utilisez [id](#) plutôt l'attribut.

**rev** sur [a](#) les éléments

**rev** sur [link](#) les éléments

Utilisez [rel](#) plutôt l'attribut, avec un terme opposé. (Par exemple, au lieu de `rev="made"`, utilisez `rel="author"`.)

**urn** sur [a](#) les éléments

**urn** sur [link](#) les éléments

Spécifiez l'identifiant persistant préféré à l'aide de l' [href](#) attribut à la place.

**accept** sur [form](#) les éléments

Utilisez plutôt l' [accept](#) attribut directement sur les [input](#) éléments.

**hreflang** sur [area](#) les éléments

**type** sur [area](#) les éléments

Ces attributs ne font rien d'utile et, pour des raisons historiques, il n'y a pas d'attributs IDL correspondants sur [area](#) les éléments. Omettez-les complètement.

**nohref** sur [area](#) les éléments

L'omission de l' [href](#) attribut est suffisante ; l' [nohref](#) attribut est inutile. Omettez-le complètement.

**profile** sur [head](#) les éléments

Inutile. Omettez-le complètement.

**manifest** sur [html](#) les éléments

Utilisez plutôt des techniciens de service. [\[SW\]](#)

### **version** sur html les éléments

Inutile. Omettez-le complètement.

### **ismap** sur input les éléments

Inutile. Omettez-le complètement. Tous input les éléments avec un type attribut dans l'état [Image Button](#) sont traités comme des images cliquables côté serveur.

### **usemap** sur input les éléments

### **usemap** sur object les éléments

Utilisez l'img élément pour les images cliquables.

### **longdesc** sur iframe les éléments

### **longdesc** sur img les éléments

Utilisez un a élément régulier pour créer un lien vers la description ou (dans le cas d'images) utilisez une [image cliquable](#) pour fournir un lien entre l'image et la description de l'image.

### **lowsrc** sur img les éléments

Utilisez une image JPEG progressive (donnée dans l'src attribut), au lieu d'utiliser deux images distinctes.

### **target** sur link les éléments

Inutile. Omettez-le complètement.

### **type** sur menu les éléments

Pour implémenter un menu contextuel personnalisé, utilisez un script pour gérer l'contextmenu événement. Pour les menus de la barre d'outils, omettez l'attribut.

### **label** sur menu les éléments

### **contextmenu** sur tous les éléments

### **onshow** sur tous les éléments

Pour implémenter un menu contextuel personnalisé, utilisez un script pour gérer l'contextmenu événement.

### **scheme** sur meta les éléments

Utilisez un seul schéma par champ ou intégrez la déclaration de schéma à la valeur.

### **archive** sur object les éléments

### **classid** sur object les éléments

### **code** sur object les éléments

### **codebase** sur object les éléments

### **codetype** sur object les éléments

Utilisez les attributs `data` et `type` pour invoquer [les plugins](#) .

#### **declare** sur `object` les éléments

Répétez l' `object` élément complètement chaque fois que la ressource doit être réutilisée.

#### **standby** sur `object` les éléments

Optimisez la ressource liée afin qu'elle se charge rapidement ou, au moins, de manière incrémentielle.

#### **typemustmatch** sur `object` les éléments

Évitez d'utiliser `object` des éléments avec des ressources non fiables.

#### **language** sur `script` les éléments (sauf comme indiqué dans la section précédente)

Omettez l'attribut pour JavaScript ; pour [les blocs de données](#) , utilisez `type` plutôt l'attribut.

#### **event** sur `script` les éléments

#### **for** sur `script` les éléments

Utilisez les mécanismes d'événements DOM pour enregistrer les écouteurs d'événements. [\[DOM\]](#)

#### **type** sur `style` les éléments (sauf comme indiqué dans la section précédente)

Omettez l'attribut pour CSS ; pour [les blocs de données](#) , utilisez `script` comme conteneur au lieu de `style`.

#### **datapagesize** sur `table` les éléments

Inutile. Omettez-le complètement.

#### **summary** sur `table` les éléments

Utilisez plutôt l'une des [techniques de description des tables](#) données dans la `table` section.

#### **abbr** sur `td` les éléments

Utilisez un texte qui commence de manière non ambiguë et concise, et incluez tout texte plus élaboré par la suite. L' `title` attribut peut également être utile pour inclure un texte plus détaillé, afin que le contenu de la cellule puisse être concis. Si c'est un titre, utilisez `th` (qui a un `abbr` attribut).

#### **axis** sur `td` et `th` éléments

Utilisez l' `scope` attribut sur le fichier `th`.

#### **scope** sur `td` les éléments

Utilisez `th` des éléments pour les cellules d'en-tête.

**datasrc** sur a, button, div, frame, iframe, img, input, label, legend, marquee, object, option, select, span, table, et textarea éléments

**datafld** sur a, button, div, fieldset, frame, iframe, img, input, label, legend, marquee, object, select, span, et textarea éléments

**dataformat** sur les

éléments button, div, input, label, legend, marquee, object, option, select, span et table

Utilisez un script et un mécanisme tel que XMLHttpRequest pour remplir la page dynamiquement. [\[XHR\]](#)

**dropzone** sur tous les éléments

Utilisez plutôt le script pour gérer les événements dragenter et dragover

**alink** sur body les éléments

**bgcolor** sur body les éléments

**bottommargin** sur body les éléments

**leftmargin** sur body les éléments

**link** sur body les éléments

**marginheight** sur body les éléments

**marginwidth** sur body les éléments

**rightmargin** sur body les éléments

**text** sur body les éléments

**topmargin** sur body les éléments

**vlink** sur body les éléments

**clear** sur br les éléments

**align** sur caption les éléments

**align** sur col les éléments

**char** sur col les éléments

**charoff** sur col les éléments

**valign** sur col les éléments

**width** sur col les éléments

**align** sur div les éléments

**compact** sur dl les éléments

**align** sur embed les éléments

**hspace** sur embed les éléments

**vspace** sur embed les éléments

**align** sur hr les éléments

**color** sur hr les éléments

**noshade** sur hr les éléments

**size** sur hr les éléments

**width** sur hr les éléments

**align** sur h1 — h6 éléments



**align**sur iframeles éléments  
**allowtransparency**sur iframeles éléments  
**frameborder**sur iframeles éléments  
**framespacing**sur iframeles éléments  
**hspace**sur iframeles éléments  
**marginheight**sur iframeles éléments  
**marginwidth**sur iframeles éléments  
**scrolling**sur iframeles éléments  
**vspace**sur iframeles éléments  
**align**sur inputles éléments  
**border**sur inputles éléments  
**hspace**sur inputles éléments  
**vspace**sur inputles éléments  
**align**sur imgles éléments  
**border**sur imgles éléments (sauf comme indiqué dans la section précédente)  
**hspace**sur imgles éléments  
**vspace**sur imgles éléments  
**align**sur legendles éléments  
**type**sur liles éléments  
**compact**sur menules éléments  
**align**sur objectles éléments  
**border**sur objectles éléments  
**hspace**sur objectles éléments  
**vspace**sur objectles éléments  
**compact**sur olles éléments  
**align**sur ples éléments  
**width**sur preles éléments  
**align**sur tableles éléments  
**bgcolor**sur tableles éléments  
**border**sur tableles éléments  
**bordercolor**sur tableles éléments  
**cellpadding**sur tableles éléments  
**cellspacing**sur tableles éléments  
**frame**sur tableles éléments  
**height**sur tableles éléments  
**rules**sur tableles éléments  
**width**sur tableles éléments  
**align**sur les éléments tbody, theadet tfoot  
**char**sur les éléments tbody, theadet tfoot  
**charoff**sur les éléments tbody, theadet tfoot

[height](#) sur les éléments [thead](#), [tbody](#) et [tfoot](#)  
[valign](#) sur les éléments [tbody](#), [thead](#) et [tfoot](#)  
[align](#) sur [td](#) et [th](#) éléments  
[bgcolor](#) sur [td](#) et [th](#) éléments  
[char](#) sur [td](#) et [th](#) éléments  
[charoff](#) sur [td](#) et [th](#) éléments  
[height](#) sur [td](#) et [th](#) éléments  
[nowrap](#) sur [td](#) et [th](#) éléments  
[valign](#) sur [td](#) et [th](#) éléments  
[width](#) sur [td](#) et [th](#) éléments  
[align](#) sur [tr](#) les éléments  
[bgcolor](#) sur [tr](#) les éléments  
[char](#) sur [tr](#) les éléments  
[charoff](#) sur [tr](#) les éléments  
[height](#) sur [tr](#) les éléments  
[valign](#) sur [tr](#) les éléments  
[compact](#) sur [ul](#) les éléments  
[type](#) sur [ul](#) les éléments  
[background](#) sur les éléments [body](#), [table](#), [thead](#), [tbody](#), [tfoot](#), [tr](#), [td](#) et [th](#)  
 Utilisez CSS à la place.

## 16.3 Exigences pour les implémentations

### 16.3.1 L' [marquee](#) élément

L' [marquee](#) élément est un élément de présentation qui anime le contenu. Les transitions et animations CSS sont un mécanisme plus approprié. [\[CSSANIMATIONS\]](#) [\[CSSTRANSITIONS\]](#)

L' [marquee](#) élément doit implémenter l' [HTMLMarqueeElement](#) interface.

```
[Exposed=Window]
```

```
interface HTMLMarqueeElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute DOMString behavior;
```

```
[CEReactions] attribute DOMString bgColor;
```

```
[CEReactions] attribute DOMString direction;
```

```
[CEReactions] attribute DOMString height;
```

```
[CEReactions] attribute unsigned long hspace;
```

```
[CEReactions] attribute long loop;
```

```
[CEReactions] attribute unsigned long scrollAmount;
```

```
[CEReactions] attribute unsigned long scrollDelay;
```

```
[CEReactions] attribute boolean trueSpeed;
```

```
[CEReactions] attribute unsigned long vspace;
```

```
[CEReactions] attribute DOMString width;
```

```
undefined start();
```

```
undefined stop();
```

```
};
```

Un [marquee](#) élément peut être **activé** ou **désactivé** . Lorsqu'il est créé, il est [activé](#) .

Lorsque la **start()** méthode est appelée, l' [marquee](#) élément doit être [activé](#) .

Lorsque la **stop()** méthode est appelée, l' [marquee](#) élément doit être [désactivé](#) .

---

L' **behavior** attribut content sur [marquee](#) les éléments est un [attribut énuméré](#) avec les mots clés suivants (tous non conformes) :

| Mot-clé | État                 |
|---------|----------------------|
| scroll  | <b>faire défiler</b> |
| slide   | <b>glisser</b>       |

| Mot-clé   | État     |
|-----------|----------|
| alternate | alterner |

La [valeur par défaut manquante](#) et [la valeur par défaut invalide](#) sont l' état [de défilement](#) .

---

L' **direction**attribut content sur [marquee](#) les éléments est un [attribut énuméré](#) avec les mots clés suivants (tous non conformes) :

| Mot-clé | État    |
|---------|---------|
| left    | gauche  |
| right   | droite  |
| up      | en haut |
| down    | bas     |

La [valeur par défaut manquante](#) et [la valeur par défaut invalide](#) sont l' état [de gauche](#) .

---

L' **truespeed**attribut content sur [marquee](#) les éléments est un [attribut booléen](#) .

---

Un [marquee](#)élément a un **intervalle de défilement de sélection** , qui s'obtient comme suit :

1. Si l'élément a un `scrollldelay`attribut et que l'analyse de sa valeur à l'aide des [règles d'analyse des entiers non négatifs](#) ne renvoie pas d'erreur, laissez *delay* être la valeur analysée. Sinon, laissez *le délai* être de 85.
2. Si l'élément n'a pas d' [truespeed](#) attribut et que la valeur *du délai* est inférieure à 60, laissez *le délai* être de 60 à la place.
3. L' [intervalle de défilement du chapiteau](#) est *delay* , interprété en millisecondes.

---

Un **marquee** élément a une **distance de défilement de sélection** , qui, si l'élément a un **scrollamount** attribut, et l'analyse de sa valeur en utilisant les [règles d'analyse des entiers non négatifs](#) ne renvoie pas d'erreur, est la valeur analysée interprétée en [pixels CSS](#) , et sinon est 6 [CSS pixels](#) .

---

Un **marquee** élément a un **marquee loop count** , qui, si l'élément a un **loop** attribut, et l'analyse de sa valeur en utilisant les [règles d'analyse des entiers](#) ne renvoie pas d'erreur ou un nombre inférieur à 1, est la valeur analysée, et sinon est -1.

L' **loop** attribut IDL, lors de l'obtention, doit renvoyer le [nombre de boucles de sélection](#) de l'élément ; et lors de la définition, si la nouvelle valeur est différente du [nombre de boucles de sélection](#) de l'élément et supérieure à zéro ou égale à -1, doit définir l' **loop** attribut de contenu de l'élément (en l'ajoutant si nécessaire) à l' [entier valide](#) qui représente la nouvelle valeur. (Les autres valeurs sont ignorées.)

Un **marquee** élément a également un **indice de boucle de courant de sélection** , qui est égal à zéro lorsque l'élément est créé.

La couche de rendu incrémentera occasionnellement l'**index de la boucle actuelle du chapiteau** , ce qui doit entraîner l'exécution des étapes suivantes :

1. Si le [nombre de boucles de sélection](#) est -1, alors retournez.
2. Incrémente de un l' [index de la boucle courante du chapiteau](#) .
3. Si l' [index de la boucle actuelle de sélection](#) est maintenant égal ou supérieur au [nombre de boucles de sélection](#) de l'élément , [désactivez](#) l' **marquee** élément.

---

Les attributs **behavior**, **direction**, **height**, **hspace**, **vspace** et **width** IDL doivent [réfléter](#) les attributs de contenu respectifs du même nom.

L' **bgColor** attribut IDL doit [réfléter](#) l' **bgcolor** attribut content.

L' **scrollAmount** attribut IDL doit [réfléter](#) l' **scrollamount** attribut content. La valeur par défaut est 6.

L' **scrollDelay**attribut IDL doit [réfléter](#) l' `scrollDelay`attribut content. La valeur par défaut est 85.

L' **trueSpeed**attribut IDL doit [réfléter](#) l' `trueSpeed`attribut content.

### 16.3.2 Cadres

L' **frameset**élément agit comme [élément de corps](#) dans les documents qui utilisent des cadres.

L' **frameset**élément doit implémenter l' `HTMLFrameSetElement` interface.

```
[Exposed=Window]

interface HTMLFrameSetElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute DOMString cols;

    [CEReactions] attribute DOMString rows;

};

HTMLFrameSetElement includes WindowEventHandlers;
```

Les attributs **cols** et **rows** IDL de l' **frameset**élément doivent [réfléter](#) les attributs de contenu respectifs du même nom.

L' **frameset**élément expose en tant [qu'attributs de contenu de gestionnaire d'événements](#) un certain nombre de [gestionnaires d'événements](#) de l' `Window`objet. Il reflète également leurs [attributs IDL de gestionnaire d'événements](#) .

Les [gestionnaires d'événements](#) de l' `Window`objet nommé par l' `Window`[ensemble de gestionnaires d'événements d'élément de corps -reflecting](#) , exposés sur l' **frameset**élément, remplacent les [gestionnaires d'événements](#) génériques par les mêmes noms normalement pris en charge par [les éléments HTML](#) .

---

L' **frame** élément a un [contenu navigable](#) similaire à l' **iframe** élément, mais rendu dans un **frameset** élément.

Un **élément frame** est dit **actif frame** lorsqu'il se trouve [dans un document](#) .

Lorsqu'un **frame** élément *element* est créé en tant [qu'élément actif frame](#) , ou devient un [élément actif frame](#) après n'en avoir pas été un, l'agent utilisateur doit exécuter ces étapes :

1. [Créez un nouvel enfant navigable](#) pour l'élément .
2. [Traitez les frame attributs](#) de *element* , avec [initialInsertion](#) défini sur true.

Lorsqu'un **frame** élément cesse d'être un [élément actif frame](#) , l'agent utilisateur doit [détruire un enfant navigable](#) compte tenu de l'élément.

Chaque fois qu'un élément avec un [contenu navigable frame](#) non nul voit son attribut défini, modifié ou supprimé, l'agent utilisateur doit [traiter les attributs](#) .srcframe

Pour **traiter les frame attributs** d'un *élément element* , avec un booléen optionnel **initialInsertion** :

1. Soit *url* le résultat de l'exécution des [étapes de traitement des attributs partagés pour iframe et frame les éléments](#) donnés *element* et *initialInsertion* .
2. Si l'URL est nulle, alors retournez.
3. Si *url* [correspond](#) `about:blank` et *initialInsertion* est vrai, alors :
  1. [Lancez un événement](#) nommé `load` à l'élément .
  2. Retour.
4. [Naviguez dans un élément iframe ou frame](#) un *élément* , une *url* et la chaîne vide.

L' **frame** élément [retarde potentiellement l'événement load](#) .

L' **frame** élément doit implémenter l' [HTMLFrameElement](#) interface.

```
[Exposed=Window]
```

```
interface HTMLFrameElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute DOMString name;
```

```

[CEReactions] attribute DOMString scrolling;

[CEReactions] attribute USVString src;

[CEReactions] attribute DOMString frameBorder;

[CEReactions] attribute USVString longDesc;

[CEReactions] attribute boolean noResize;

readonly attribute Document? contentDocument;

readonly attribute WindowProxy? contentWindow;


[CEReactions] attribute [LegacyNullToEmptyString] DOMString
marginHeight;

[CEReactions] attribute [LegacyNullToEmptyString] DOMString
marginWidth;

};

```

Les attributs **name**, **scrolling** et **src** IDL de l' **frame** élément doivent [refléter](#) les attributs de contenu respectifs du même nom. Pour les besoins de la réflexion, l' attribut contenu **frame** de l'élément **src** est défini comme contenant une [URL](#) .

L' **frameBorder** attribut IDL de l' **frame** élément doit [refléter](#) l'attribut contenu de l'élément `frameborder`.

L' **longDesc** attribut IDL de l' **frame** élément doit [refléter](#) l'attribut contenu de l'élément `longdesc`, qui, à des fins de réflexion, est défini comme contenant une [URL](#) .

L' **noResize** attribut IDL de l' **frame** élément doit [refléter](#) l'attribut contenu de l'élément `noresize`.

L' **marginHeight** attribut IDL de l' **frame** élément doit [refléter](#) l'attribut contenu de l'élément `marginheight`.

L' **marginWidth** attribut IDL de l' **frame** élément doit [refléter](#) l'attribut contenu de l'élément `marginwidth`.



Les `contentDocument` étapes du getter consistent à retourner [ce](#) document de [contenu](#) .

Les `contentWindow` étapes du getter consistent à retourner [cette](#) fenêtre de [contenu](#) .

### 16.3.3 Autres éléments, attributs et API

Les agents utilisateurs doivent traiter [acronym](#) les éléments d'une manière équivalente aux [abbr](#) éléments en termes de sémantique et à des fins de rendu.

---

```
partial interface HTMLAnchorElement {  
  
  [CEReactions] attribute DOMString coords;  
  
  [CEReactions] attribute DOMString charset;  
  
  [CEReactions] attribute DOMString name;  
  
  [CEReactions] attribute DOMString rev;  
  
  [CEReactions] attribute DOMString shape;  
  
};
```

Les attributs `coords`, `charset`, , et IDL de l' élément doivent [refléter](#) `name` les attributs de contenu respectifs du même nom. `rev` `shape` `a`

---

```
partial interface HTMLAreaElement {  
  
  [CEReactions] attribute boolean noHref;  
  
};
```

L' **noHref**attribut IDL de l' **area**élément doit [réfléter](#) l'attribut content de l'élément **nohref**.

---

```
partial interface HTMLBodyElement {  
  
    [CEReactions] attribute [LegacyNullToEmptyString] DOMString  
    text;  
  
    [CEReactions] attribute [LegacyNullToEmptyString] DOMString  
    link;  
  
    [CEReactions] attribute [LegacyNullToEmptyString] DOMString  
    vLink;  
  
    [CEReactions] attribute [LegacyNullToEmptyString] DOMString  
    aLink;  
  
    [CEReactions] attribute [LegacyNullToEmptyString] DOMString  
    bgColor;  
  
    [CEReactions] attribute DOMString background;  
  
};
```

L' **text**attribut IDL de l' **body**élément doit [réfléter](#) l'attribut content de l'élément **text**.

L' **link**attribut IDL de l' **body**élément doit [réfléter](#) l'attribut content de l'élément **link**.

L' **aLink**attribut IDL de l' **body**élément doit [réfléter](#) l'attribut content de l'élément **alink**.

L' **vLink**attribut IDL de l' **body**élément doit [réfléter](#) l'attribut content de l'élément **vlink**.

L' **bgColor** attribut IDL de l' **body**élément doit [réfléter](#) l'attribut content de l'élément **bgcolor**.

L' **background**attribut IDL de l' **body** élément doit [réfléter](#) l'attribut content de l'élément **background** . (Le **background** contenu *n'est pas* défini pour contenir une [URL](#) , malgré les règles concernant sa gestion dans la section Rendu ci-dessus.)

---

```
partial interface HTMLBRElement {
```

```
  [CEReactions] attribute DOMString clear;
```

```
};
```

L' **clear** attribut IDL de l' [br](#) élément doit [réfléter](#) l'attribut content du même nom.

---

```
partial interface HTMLTableCaptionElement {
```

```
  [CEReactions] attribute DOMString align;
```

```
};
```

L' **align** attribut IDL de l' [caption](#) élément doit [réfléter](#) l'attribut content du même nom.

---

```
partial interface HTMLTableColElement {
```

```
  [CEReactions] attribute DOMString align;
```

```
  [CEReactions] attribute DOMString ch;
```

```
  [CEReactions] attribute DOMString chOff;
```

```
  [CEReactions] attribute DOMString vAlign;
```

```
  [CEReactions] attribute DOMString width;
```

```
};
```

Les attributs **align** et **width**IDL de l' col élément doivent [réfléter](#) les attributs de contenu respectifs du même nom.

L' **ch**attribut IDL de l' col élément doit [réfléter](#) l'attribut content de l'élément char.

L' **chOff** attribut IDL de l' col élément doit [réfléter](#) l'attribut content de l'élément charoff.

L' **vAlign** attribut IDL de l' col élément doit [réfléter](#) l'attribut content de l'élément valign.

---

Les agents utilisateurs doivent traiter dir les éléments d'une manière équivalente aux ul éléments en termes de sémantique et à des fins de rendu.

L' dir élément doit implémenter l' HTMLDirectoryElement interface.

```
[Exposed=Window]

interface HTMLDirectoryElement : HTMLElement {

    [HTMLConstructor] constructor();

    [CEReactions] attribute boolean compact;

};
```

L' **compact**attribut IDL de l' dir élément doit [réfléter](#) l'attribut content du même nom.

---

```
partial interface HTMLDivElement {

    [CEReactions] attribute DOMString align;

};
```

L' **align**attribut IDL de l' div élément doit [réfléter](#) l'attribut content du même nom.

---

```
partial interface HTMLDListElement {  
  
    [CEReactions] attribute boolean compact;  
  
};
```

L' **compact** attribut IDL de l' [dl](#) élément doit [réfléter](#) l'attribut content du même nom.

---

```
partial interface HTMLEmbedElement {  
  
    [CEReactions] attribute DOMString align;  
  
    [CEReactions] attribute DOMString name;  
  
};
```

Les attributs **name** et **align** IDL de l' [embed](#) élément doivent [réfléter](#) les attributs de contenu respectifs du même nom.

---

L' [font](#) élément doit implémenter l' [HTMLFontElement](#) interface.

```
[Exposed=Window]  
  
interface HTMLFontElement : HTMLElement {  
  
    [HTMLConstructor] constructor();  
  
  
    [CEReactions] attribute [LegacyNullToEmptyString] DOMString  
    color;
```

```
[CEReactions] attribute DOMString face;
```

```
[CEReactions] attribute DOMString size;
```

```
};
```

Les attributs **color**, **face** et **size** IDL de l' [font](#) élément doivent [refléter](#) les attributs de contenu respectifs du même nom.

---

```
partial interface HTMLHeadingElement {
```

```
[CEReactions] attribute DOMString align;
```

```
};
```

L' **align** attribut IDL des éléments [h1](#)–[h6](#) doit [refléter](#) l'attribut content du même nom.

---

*L' **profile** attribut IDL sur [head](#) les éléments (avec l' [HTMLHeadElement](#) interface) est intentionnellement omis. À moins que cela ne soit requis par [une autre spécification applicable](#) , les implémentations ne prendraient donc pas en charge cet attribut. (Il est mentionné ici tel qu'il était défini dans une version précédente de DOM .)*

---

```
partial interface HTMLHRElement {
```

```
[CEReactions] attribute DOMString align;
```

```
[CEReactions] attribute DOMString color;
```

```
[CEReactions] attribute boolean noShade;
```

```
[CEReactions] attribute DOMString size;
```

```
[CEReactions] attribute DOMString width;
```

```
};
```

Les attributs **align**, **color**, **size** et **width** IDL de l' **hr** élément doivent [réfléter](#) les attributs de contenu respectifs du même nom.

L' **noShade** attribut IDL de l' **hr** élément doit [réfléter](#) l'attribut contenu de l'élément **noshade**.

---

```
partial interface HTMLHtmlElement {
```

```
[CEReactions] attribute DOMString version;
```

```
};
```

L' **version** attribut IDL de l' **html** élément doit [réfléter](#) l'attribut contenu du même nom.

---

```
partial interface HTMLIFrameElement {
```

```
[CEReactions] attribute DOMString align;
```

```
[CEReactions] attribute DOMString scrolling;
```

```
[CEReactions] attribute DOMString frameBorder;
```

```
[CEReactions] attribute USVString longDesc;
```

```
[CEReactions] attribute [LegacyNullToEmptyString] DOMString
```

```
marginHeight;
```

```

[CEReactions] attribute [LegacyNullToEmptyString] DOMString
marginWidth;

};

```

Les attributs **align** et **scrolling**IDL de l' `iframe` élément doivent [refléter](#) les attributs de contenu respectifs du même nom.

L' **frameBorder**attribut IDL de l' `iframe` élément doit [refléter](#) l'attribut content de l'élément `frameborder`.

L' **longDesc**attribut IDL de l' `iframe` élément doit [refléter](#) l'attribut content de l'élément `longdesc`, qui, à des fins de réflexion, est défini comme contenant une [URL](#) .

L' **marginHeight**attribut IDL de l' `iframe`élément doit [refléter](#) l'attribut content de l'élément `marginheight`.

L' **marginWidth**attribut IDL de l' `iframe` élément doit [refléter](#) l'attribut content de l'élément `marginwidth`.

```

partial interface HTMLImageElement {

[CEReactions] attribute DOMString name;

[CEReactions] attribute USVString lowsrc;

[CEReactions] attribute DOMString align;

[CEReactions] attribute unsigned long hspace;

[CEReactions] attribute unsigned long vspace;

[CEReactions] attribute USVString longDesc;

[CEReactions] attribute [LegacyNullToEmptyString] DOMString
border;

```



```
};
```

Les attributs **name**, **align**, , et IDL de l' élément doivent [refléter](#) **border** les attributs de contenu respectifs du même nom. **hspace** **vspace** **img**

L' **longDesc** attribut IDL de l' **img** élément doit [refléter](#) l'attribut contenu de l'élément **longdesc**, qui, à des fins de réflexion, est défini comme contenant une [URL](#) .

L' **lowsrc** attribut IDL de l' **img** élément doit [refléter](#) l'attribut contenu de l'élément **lowsrc**, qui, à des fins de réflexion, est défini comme contenant une [URL](#) .

---

```
partial interface HTMLInputElement {
```

```
    [CEReactions] attribute DOMString align;
```

```
    [CEReactions] attribute DOMString useMap;
```

```
};
```

L' **align** attribut IDL de l' **input** élément doit [refléter](#) l'attribut contenu du même nom.

L' **useMap** attribut IDL de l' **input** élément doit [refléter](#) l'attribut contenu de l'élément **usemap**.

---

```
partial interface HTMLLegendElement {
```

```
    [CEReactions] attribute DOMString align;
```

```
};
```

L' **align** attribut IDL de l' **legend** élément doit [refléter](#) l'attribut contenu du même nom.

---

```
partial interface HTMLLIElement {
    [CEReactions] attribute DOMString type;
};
```

L' **type** attribut IDL de l' **li** élément doit [réfléter](#) l'attribut content du même nom.

---

```
partial interface HTMLLinkElement {
    [CEReactions] attribute DOMString charset;
    [CEReactions] attribute DOMString rev;
    [CEReactions] attribute DOMString target;
};
```

Les attributs **charset**, **rev** et **target** IDL de l' **link** élément doivent [réfléter](#) les attributs de contenu respectifs du même nom.

---

Les agents utilisateurs doivent traiter [listing](#) les éléments d'une manière équivalente aux [pre](#) éléments en termes de sémantique et à des fins de rendu.

---

```
partial interface HTMLMenuElement {
    [CEReactions] attribute boolean compact;
};
```

L' **compact** attribut IDL de l' menu élément doit [refléter](#) l'attribut content du même nom.

---

```
partial interface HTMLMetaElement {  
  
    [CEReactions] attribute DOMString scheme;  
  
};
```

Les agents utilisateurs peuvent traiter l' scheme attribut content sur l' meta élément comme une extension de l'attribut content de l'élément name lors du traitement d'un meta élément avec un name attribut dont la valeur est celle que l'agent utilisateur reconnaît comme supportant l' scheme attribut.

Les agents utilisateurs sont encouragés à ignorer l' scheme attribut et à traiter à la place la valeur donnée au nom de la métadonnée comme si elle avait été spécifiée pour chaque valeur attendue de l' scheme attribut.

Par exemple, si l'agent utilisateur agit sur meta des éléments avec name des attributs ayant la valeur "eGMS.subject.keyword", et sait que l' scheme attribut est utilisé avec ce nom de métadonnées, alors il pourrait prendre l' scheme attribut en compte, agissant comme s'il s'agissait d'un extension de l' name attribut. Ainsi, les deux meta éléments suivants pourraient être traités comme deux éléments donnant des valeurs pour deux noms de métadonnées différents, l'un consistant en une combinaison de "eGMS.subject.keyword" et "LGCL", et l'autre consistant en une combinaison de "eGMS.subject. mot-clé" et "ORLY":

```
<!-- this markup is invalid -->  
  
<meta name="eGMS.subject.keyword" scheme="LGCL" content="Abandoned  
vehicles">  
  
<meta name="eGMS.subject.keyword" scheme="ORLY" content="Mah car:  
kthxbye">
```

Le traitement suggéré de ce balisage, cependant, serait équivalent à ce qui suit :

```
<meta name="eGMS.subject.keyword" content="Abandoned vehicles">  
  
<meta name="eGMS.subject.keyword" content="Mah car: kthxbye">
```

L' **scheme** attribut IDL de l' meta élément doit [refléter](#) l'attribut content du même nom.

---

```

partial interface HTMLObjectElement {

    [CEReactions] attribute DOMString align;

    [CEReactions] attribute DOMString archive;

    [CEReactions] attribute DOMString code;

    [CEReactions] attribute boolean declare;

    [CEReactions] attribute unsigned long hspace;

    [CEReactions] attribute DOMString standby;

    [CEReactions] attribute unsigned long vspace;

    [CEReactions] attribute DOMString codeBase;

    [CEReactions] attribute DOMString codeType;

    [CEReactions] attribute DOMString useMap;

    [CEReactions] attribute [LegacyNullToEmptyString] DOMString
    border;

};

```

Les attributs **align**, **archive**, **border**, **code**, **declare**, **hspace**, **standby** et **vspace** IDL de l' **object** élément doivent **refléter** les attributs de contenu respectifs du même nom.

L' **codeBase** attribut IDL de l' **object** élément doit **refléter** l'attribut contenu de l'élément **codebase**, qui, à des fins de réflexion, est défini comme contenant une **URL** .

L' **codeType** attribut IDL de l' **object** élément doit **refléter** l'attribut contenu de l'élément **codetype**.



L' **useMap** attribut IDL doit **refléter** l' **usemap** attribut contenu.

---

```
partial interface HTMLListElement {
```

```
  [CEReactions] attribute boolean compact;
```

```
};
```

L' **compact** attribut IDL de l' [ol](#) élément doit [réfléter](#) l'attribut content du même nom.

---

```
partial interface HTMLParagraphElement {
```

```
  [CEReactions] attribute DOMString align;
```

```
};
```

L' **align** attribut IDL de l' [p](#) élément doit [réfléter](#) l'attribut content du même nom.

---

L' [param](#) élément doit implémenter l' [HTMLParamElement](#) interface.

```
[Exposed=Window]
```

```
interface HTMLParamElement : HTMLElement {
```

```
  [HTMLConstructor] constructor();
```

```
  [CEReactions] attribute DOMString name;
```

```
  [CEReactions] attribute DOMString value;
```

```
  [CEReactions] attribute DOMString type;
```

```
  [CEReactions] attribute DOMString valueType;
```

```
};
```

Les attributs **name**, **value** et **type** IDL de l' param élément doivent [réfléter](#) les attributs de contenu respectifs du même nom.

L' **value** attribut IDL de l' param élément doit [réfléter](#) l'attribut contenu de l'élément `value`.

---

Les agents utilisateurs doivent traiter plaintext les éléments d'une manière équivalente aux pre éléments en termes de sémantique et à des fins de rendu. (L'analyseur a cependant un comportement spécial pour cet élément.)

---

```
partial interface HTMLPreElement {  
  
    [CEReactions] attribute long width;  
  
};
```

L' **width** attribut IDL de l' pre élément doit [réfléter](#) l'attribut contenu du même nom.

---

```
partial interface HTMLStyleElement {  
  
    [CEReactions] attribute DOMString type;  
  
};
```

L' **type** attribut IDL de l' style élément doit [réfléter](#) l'attribut contenu de l'élément type.

---

```

partial interface HTMLScriptElement {

  [CEReactions] attribute DOMString charset;

  [CEReactions] attribute DOMString event;

  [CEReactions] attribute DOMString htmlFor;

};

```

Les attributs **charset** et **event** IDL de l' script élément doivent refléter les attributs de contenu respectifs du même nom.

L' **htmlFor** attribut IDL de l' script élément doit refléter l'attribut contenu de l'élément for.

```

partial interface HTMLTableElement {

  [CEReactions] attribute DOMString align;

  [CEReactions] attribute DOMString border;

  [CEReactions] attribute DOMString frame;

  [CEReactions] attribute DOMString rules;

  [CEReactions] attribute DOMString summary;

  [CEReactions] attribute DOMString width;

  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
  bgColor;

  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
  cellPadding;

```

```

[CEReactions] attribute [LegacyNullToEmptyString] DOMString
cellSpacing;

};

```

Les attributs IDL **align**, **border**, **frame**, **summary**, **rules**, et , de l' élément doivent [refléter](#) les attributs de contenu respectifs du même nom. **width**[table](#)

L' **bgColor** attribut IDL de l' [table](#) élément doit [refléter](#) l'attribut content de l'élément [bgcolor](#).

L' **cellPadding**attribut IDL de l' [table](#) élément doit [refléter](#) l'attribut content de l'élément [cellpadding](#).

L' **cellSpacing**attribut IDL de l' [table](#) élément doit [refléter](#) l'attribut content de l'élément [cellspacing](#).

```

partial interface HTMLTableSectionElement {

[CEReactions] attribute DOMString align;

[CEReactions] attribute DOMString ch;

[CEReactions] attribute DOMString chOff;

[CEReactions] attribute DOMString vAlign;

};

```

L' **align**attribut IDL des éléments [tbody](#), [thead](#)et [tfoot](#)doit [refléter](#) l'attribut content du même nom.

L' **ch** attribut IDL des éléments [tbody](#), [thead](#)et [tfoot](#)doit [refléter](#) les attributs de contenu des éléments [char](#).

L' **chOff**attribut IDL des éléments [tbody](#), [thead](#)et [tfoot](#)doit [refléter](#) les attributs de contenu des éléments [charoff](#).

L' **vAlign**attribut IDL des éléments [tbody](#), [thead](#)et [tfoot](#)doit [refléter](#) les attributs de contenu des éléments [valign](#).



---

```
partial interface HTMLTableCellElement {
```

```
  [CEReactions] attribute DOMString align;
```

```
  [CEReactions] attribute DOMString axis;
```

```
  [CEReactions] attribute DOMString height;
```

```
  [CEReactions] attribute DOMString width;
```

```
  [CEReactions] attribute DOMString ch;
```

```
  [CEReactions] attribute DOMString chOff;
```

```
  [CEReactions] attribute boolean noWrap;
```

```
  [CEReactions] attribute DOMString vAlign;
```

```
  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
```

```
  bgColor;
```

```
};
```

Les attributs **align**, **axis**, **height** et **width** IDL des éléments **td** et **th** doivent [réfléter](#) les attributs de contenu respectifs du même nom.

L' **ch** attribut IDL des éléments **td** et **th** doit [réfléter](#) les attributs de contenu des éléments **char**.

L' **chOff** attribut IDL des éléments **td** et **th** doit [réfléter](#) les attributs de contenu des éléments **charoff**.

L' **noWrap** attribut IDL des éléments **td** et **th** doit [réfléter](#) les attributs de contenu des éléments **nowrap**.

L' **vAlign** attribut IDL des éléments **td** et **th** doit [réfléter](#) les attributs de contenu des éléments **valign**.

L' **bgColor** attribut IDL des éléments td et th doit [réfléter](#) les attributs de contenu des éléments bgcolor.

---

```
partial interface HTMLTableRowElement {  
  
    [CEReactions] attribute DOMString align;  
  
    [CEReactions] attribute DOMString ch;  
  
    [CEReactions] attribute DOMString chOff;  
  
    [CEReactions] attribute DOMString vAlign;  
  
    [CEReactions] attribute [LegacyNullToEmptyString] DOMString  
    bgColor;  
  
};
```

L' **align** attribut IDL de l' tr élément doit [réfléter](#) l'attribut content du même nom.

L' **ch** attribut IDL de l' tr élément doit [réfléter](#) l'attribut content de l'élément char.

L' **chOff** attribut IDL de l' tr élément doit [réfléter](#) l'attribut content de l'élément charoff.

L' **vAlign** attribut IDL de l' tr élément doit [réfléter](#) l'attribut content de l'élément valign.

L' **bgColor** attribut IDL de l' tr élément doit [réfléter](#) l'attribut content de l'élément bgcolor.

---

```
partial interface HTMLUListElement {  
  
    [CEReactions] attribute boolean compact;
```

```
[CEReactions] attribute DOMString type;
```

```
};
```

Les attributs **compact** et **type**IDL de l'ulélément doivent refléter les attributs de contenu respectifs du même nom.

---

Les agents utilisateurs doivent traiter xmp les éléments d'une manière équivalente aux pre éléments en termes de sémantique et à des fins de rendu. (L'analyseur a cependant un comportement spécial pour cet élément.)

---

```
partial interface Document {
```

```
  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
```

```
  fgColor;
```

```
  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
```

```
  linkColor;
```

```
  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
```

```
  vlinkColor;
```

```
  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
```

```
  alinkColor;
```

```
  [CEReactions] attribute [LegacyNullToEmptyString] DOMString
```

```
  bgColor;
```

```
  [SameObject] readonly attribute HTMLCollection anchors;
```

```
  [SameObject] readonly attribute HTMLCollection applets;
```

```
undefined clear\(\) ;
```

```
undefined captureEvents\(\) ;
```

```
undefined releaseEvents\(\) ;
```

```
[SameObject] readonly attribute HTMLAllCollection all;
```

```
};
```

Les attributs de l' [Document](#) objet répertoriés dans la première colonne du tableau suivant doivent [refléter](#) l'attribut de contenu sur [l'élément body](#) avec le nom donné dans la cellule correspondante de la deuxième colonne sur la même ligne, si [l'élément body](#) est un [body](#) élément (par opposition à un [frameset](#) élément). Lorsqu'il n'y a pas [d'élément body](#) ou s'il s'agit d'un [frameset](#) élément, les attributs doivent à la place renvoyer la chaîne vide lors de l'obtention et ne rien faire lors de la définition.

| Attribut IDL               | Attribut de contenu     |
|----------------------------|-------------------------|
| <a href="#">fgColor</a>    | <a href="#">text</a>    |
| <a href="#">linkColor</a>  | <a href="#">link</a>    |
| <a href="#">vlinkColor</a> | <a href="#">vlink</a>   |
| <a href="#">alinkColor</a> | <a href="#">alink</a>   |
| <a href="#">bgColor</a>    | <a href="#">bgcolor</a> |

---

L' [anchors](#) attribut doit renvoyer un [HTMLCollection](#) enraciné au [Document](#) nœud, dont le filtre ne correspond qu'aux [a](#) éléments avec [name](#) des attributs.

L' [applets](#) attribut doit renvoyer un [HTMLCollection](#) enraciné au [Document](#) nœud, dont le filtre ne correspond à rien. (Il existe pour des raisons historiques.)

Les méthodes [clear\(\)](#), [captureEvents\(\)](#) et [releaseEvents\(\)](#) ne doivent rien faire.

---

L' [all](#) attribut doit renvoyer un [HTMLAllCollection](#) enraciné au [Document](#) nœud, dont le filtre correspond à tous les éléments.

---

```
partial interface Window {
```

```
    undefined captureEvents\(\) ;
```

```
    undefined releaseEvents\(\) ;
```

```
[Replaceable, SameObject] readonly attribute External external;
```

```
};
```

Les méthodes `captureEvents()` et `releaseEvents()` ne doivent rien faire.

L' `external` attribut de l' `Window` interface doit renvoyer une instance de l' `External` interface :

```
[Exposed=Window]
```

```
interface External {
```

```
    undefined AddSearchProvider\(\) ;
```

```
    undefined IsSearchProviderInstalled\(\) ;
```

```
};
```

Les méthodes `AddSearchProvider()` et `IsSearchProviderInstalled()` ne doivent rien faire.

## 1. [17 Considérations relatives à l'IANA](#)

1. [17.1\\_text/html](#)
2. [17.2\\_multipart/x-mixed-replace](#)
3. [17.3\\_application/xhtml+xml](#)
4. [17.4\\_text/ping](#)
5. [17.5\\_application/microdata+json](#)
6. [17.6\\_text/event-stream](#)
7. [17.7\\_web+préfixe de schéma](#)

## 17 Considérations relatives à l'IANA

### 17.1 `text/html`

Cet enregistrement est destiné à l'examen de la communauté et sera soumis à l'IESG pour examen, approbation et enregistrement auprès de l'IANA.

**Tapez le nom :**

texte

**Nom du sous-type :**

html

**Paramètres requis :**

Aucun paramètre requis

**Paramètres facultatifs :**

`charset`

Le `charset` paramètre peut être fourni pour spécifier le [codage de caractères du document](#), remplaçant toute [déclaration de codage de caractères](#) dans le document autre qu'une marque d'ordre d'octet (BOM). La valeur du paramètre doit être une correspondance [ASCII non sensible à la casse](#) pour la chaîne " utf-8". [\[CODAGE\]](#)

**Considérations relatives à l'encodage :**

8 bits (voir la section sur [les déclarations d'encodage de caractères](#) )

**Considérations de sécurité :**

Des romans entiers ont été écrits sur les considérations de sécurité qui s'appliquent aux documents HTML. Beaucoup sont répertoriés dans ce document, auquel le lecteur est renvoyé pour plus de détails. Cependant, certaines préoccupations générales méritent d'être mentionnées ici :

HTML est un langage de script et possède un grand nombre d'API (dont certaines sont décrites dans ce document). Le script peut exposer l'utilisateur à des risques potentiels de fuite d'informations, de fuite d'informations d'identification, d'attaques de scripts intersites, de falsifications de requêtes intersites et à une foule d'autres problèmes. Alors que les conceptions de cette spécification sont censées être sûres si elles sont correctement implémentées, une implémentation complète est une entreprise massive et, comme pour tout logiciel, les agents utilisateurs sont susceptibles d'avoir des bogues de sécurité.

Même sans script, il existe des fonctionnalités spécifiques en HTML qui, pour des raisons historiques, sont nécessaires pour une large compatibilité avec le contenu hérité, mais qui exposent l'utilisateur à de malheureux problèmes de sécurité. En particulier, l' `img` élément peut être utilisé en conjonction avec d'autres fonctionnalités comme moyen d'effectuer une analyse de port à partir de l'emplacement de l'utilisateur sur Internet. Cela peut exposer des

topologies de réseau local que l'attaquant ne serait autrement pas en mesure de déterminer.

HTML s'appuie sur un schéma de compartimentation parfois connu sous le nom de *politique de même origine*. Une [origine](#) consiste dans la plupart des cas en toutes les pages servies depuis le même hôte, sur le même port, en utilisant le même protocole.

Il est donc essentiel de s'assurer que tout contenu non fiable qui fait partie d'un site soit hébergé sur une [origine](#) différente de tout contenu sensible sur ce site. Un contenu non fiable peut facilement usurper n'importe quelle autre page sur la même origine, lire des données de cette origine, provoquer l'exécution de scripts de cette origine, soumettre des formulaires vers et depuis cette origine même s'ils sont protégés contre les attaques de falsification de requêtes intersites par des jetons uniques, et utiliser les ressources de tiers exposées ou les droits accordés à cette origine.

### **Considérations d'interopérabilité :**

Les règles de traitement du contenu conforme et non conforme sont définies dans la présente spécification.

### **Spécification publiée :**

Ce document est la spécification pertinente. L'étiquetage d'une ressource avec le [text/html](#) type affirme que la ressource est un [document HTML](#) utilisant [la syntaxe HTML](#).

### **Applications qui utilisent ce type de média :**

Navigateurs Web, outils de traitement de contenu Web, outils auteurs HTML, moteurs de recherche, validateurs.

### **Informations Complémentaires:**

#### **Numéro(s) magique(s) :**

Aucune séquence d'octets ne peut identifier de manière unique un document HTML. Plus d'informations sur la détection des documents HTML sont disponibles dans *MIME Sniffing*. [\[MIMESNIFF\]](#)

#### **Extension(s) de fichier :**

"html" et "htm" sont couramment, mais certainement pas exclusivement, utilisés comme extension pour les documents HTML.

#### **Code(s) de type de fichier Macintosh :**

TEXT

### **Personne et adresse e-mail à contacter pour plus d'informations :**

Ian Hickson <ian@hixie.ch>

### **Utilisation prévue :**

Commun

### **Restrictions d'utilisation :**

Aucune restriction ne s'applique.

### **Auteur:**

Ian Hickson <ian@hixie.ch>

**Changer de contrôleur :**

W3C

[Les fragments](#) utilisés avec [text/html](#) les ressources font soit référence à la [partie indiquée](#) du correspondant [Document](#), soit fournissent des informations d'état pour les scripts dans la page.

## 17.2 [multipart/x-mixed-replace](#)

Cet enregistrement est destiné à l'examen de la communauté et sera soumis à l'IESG pour examen, approbation et enregistrement auprès de l'IANA.

**Tapez le nom :**

en plusieurs parties

**Nom du sous-type :**

x-mélange-remplacement

**Paramètres requis :**

- `boundary`(défini dans RFC2046) [\[RFC2046\]](#)

**Paramètres facultatifs :**

Aucun paramètre facultatif.

**Considérations relatives à l'encodage :**

binaire

**Considérations de sécurité :**

Les sous-ressources d'une [multipart/x-mixed-replace](#) ressource peuvent être de n'importe quel type, y compris des types avec des implications de sécurité non triviales telles que [text/html](#).

**Considérations d'interopérabilité :**

Aucun.

**Spécification publiée :**

Cette spécification décrit les règles de traitement pour les navigateurs Web. Les exigences de conformité pour générer des ressources avec ce type sont les mêmes que pour [multipart/mixed](#). [\[RFC2046\]](#)

**Applications qui utilisent ce type de média :**

Ce type est destiné à être utilisé dans les ressources générées par les serveurs Web, pour être consommées par les navigateurs Web.

**Informations Complémentaires:**

**Numéro(s) magique(s) :**



Aucune séquence d'octets ne peut identifier de manière unique une [multipart/x-mixed-replace](#) ressource.

**Extension(s) de fichier :**

Aucune extension de fichier spécifique n'est recommandée pour ce type.

**Code(s) de type de fichier Macintosh :**

Aucun code de type de fichier Macintosh spécifique n'est recommandé pour ce type.

**Personne et adresse e-mail à contacter pour plus d'informations :**

Ian Hickson <[ian@hixie.ch](mailto:ian@hixie.ch)>

**Utilisation prévue :**

Commun

**Restrictions d'utilisation :**

Aucune restriction ne s'applique.

**Auteur:**

Ian Hickson <[ian@hixie.ch](mailto:ian@hixie.ch)>

**Changer de contrôleur :**

W3C

[Les fragments](#) utilisés avec [multipart/x-mixed-replace](#) les ressources s'appliquent à chaque partie du corps tel que défini par le type utilisé par cette partie du corps.

## 17.3 [application/xhtml+xml](#)

Cet enregistrement est destiné à l'examen de la communauté et sera soumis à l'IESG pour examen, approbation et enregistrement auprès de l'IANA.

**Tapez le nom :**

application

**Nom du sous-type :**

xhtml+xml

**Paramètres requis :**

Identique à [application/xml](#) [\[RFC7303\]](#)

**Paramètres facultatifs :**

Identique à [application/xml](#) [\[RFC7303\]](#)

**Considérations relatives à l'encodage :**

Identique à [application/xml](#) [\[RFC7303\]](#)

**Considérations de sécurité :**

Identique à [application/xml](#) [RFC7303]

**Considérations d'interopérabilité :**

Identique à [application/xml](#) [RFC7303]

**Spécification publiée :**

L'étiquetage d'une ressource avec le [application/xhtml+xml](#) type affirme que la ressource est un document XML qui a probablement un [élément de document](#) de l' [espace de noms HTML](#) . Ainsi, les spécifications pertinentes sont XML , Namespaces in XML et cette spécification. [XML] [XMLNS]

**Applications qui utilisent ce type de média :**

Identique à [application/xml](#) [RFC7303]

**Informations Complémentaires:**

**Numéro(s) magique(s) :**

Identique à [application/xml](#) [RFC7303]

**Extension(s) de fichier :**

" xhtml " et " xht " sont parfois utilisés comme extensions pour les ressources XML qui ont un [élément document](#) de l' [espace de noms HTML](#) .

**Code(s) de type de fichier Macintosh :**

TEXT

**Personne et adresse e-mail à contacter pour plus d'informations :**

Ian Hickson <ian@hixie.ch>

**Utilisation prévue :**

Commun

**Restrictions d'utilisation :**

Aucune restriction ne s'applique.

**Auteur:**

Ian Hickson <ian@hixie.ch>

**Changer de contrôleur :**

W3C

[Les fragments](#) utilisés avec [application/xhtml+xml](#) les ressources ont la même sémantique qu'avec n'importe quel [type XML MIME](#) . [RFC7303]

## 17.4 [text/ping](#)

Cet enregistrement est destiné à l'examen de la communauté et sera soumis à l'IESG pour examen, approbation et enregistrement auprès de l'IANA.

**Tapez le nom :**

texte

**Nom du sous-type :**

ping

**Paramètres requis :**

Aucun paramètre

**Paramètres facultatifs :**

`charset`

Le `charset` paramètre peut être fourni. La valeur du paramètre doit être " `utf-8`". Ce paramètre ne sert à rien ; il n'est autorisé que pour la compatibilité avec les serveurs hérités.

**Considérations relatives à l'encodage :**

N'est pas applicable.

**Considérations de sécurité :**

S'il est utilisé exclusivement de la manière décrite dans le contexte de [l'audit des liens hypertexte](#) , ce type n'introduit aucun nouveau problème de sécurité.

**Considérations d'interopérabilité :**

Les règles applicables à ce type sont définies dans la présente spécification.

**Spécification publiée :**

Ce document est la spécification pertinente.

**Applications qui utilisent ce type de média :**

Navigateurs Web.

**Informations Complémentaires:**

**Numéro(s) magique(s) :**

`text/ping` les ressources sont toujours constituées des quatre octets 0x50 0x49 0x4E 0x47 ( `` PING`` ).

**Extension(s) de fichier :**

Aucune extension de fichier spécifique n'est recommandée pour ce type.

**Code(s) de type de fichier Macintosh :**

Aucun code de type de fichier Macintosh spécifique n'est recommandé pour ce type.

**Personne et adresse e-mail à contacter pour plus d'informations :**

Ian Hickson <ian@hixie.ch>

**Utilisation prévue :**

Commun

**Restrictions d'utilisation :**

Uniquement destiné à être utilisé avec les requêtes HTTP POST générées dans le cadre du traitement de l' `ping` attribut par un navigateur Web.

**Auteur:**

Ian Hickson <ian@hixie.ch>

**Changer de contrôleur :**

W3C

[Les fragments](#) n'ont aucun sens avec [text/ping](#) les ressources.

## 17.5 [application/microdata+json](#)

Cet enregistrement est destiné à l'examen de la communauté et sera soumis à l'IESG pour examen, approbation et enregistrement auprès de l'IANA.

**Tapez le nom :**

application

**Nom du sous-type :**

microdonnées+json

**Paramètres requis :**

Identique à [application/json](#) [\[JSON\]](#)

**Paramètres facultatifs :**

Identique à [application/json](#) [\[JSON\]](#)

**Considérations relatives à l'encodage :**

8 bits (toujours UTF-8)

**Considérations de sécurité :**

Identique à [application/json](#) [\[JSON\]](#)

**Considérations d'interopérabilité :**

Identique à [application/json](#) [\[JSON\]](#)

**Spécification publiée :**

L'étiquetage d'une ressource avec le [application/microdata+json](#) type affirme que la ressource est un texte JSON qui se compose d'un objet avec une seule entrée appelée " `items` " consistant en un tableau d'entrées, dont chacune se compose d'un objet avec une entrée appelée " `id` " dont la valeur est un chaîne, une entrée appelée " `type` " dont la valeur est une autre chaîne, et une entrée appelée " `properties` " dont la valeur est un objet dont les entrées ont chacune une valeur constituée d'un tableau d'objets ou de chaînes, les objets étant de la même forme que les objets dans l' `items` entrée " " susmentionnée. Ainsi, les spécifications pertinentes sont *JSON* et cette spécification. [\[JSON\]](#)

**Applications qui utilisent ce type de média :**

Les applications qui transfèrent des données destinées à être utilisées avec la fonctionnalité de microdonnées HTML, en particulier dans le contexte du glisser-déposer, constituent la principale classe d'applications pour ce type.

**Informations Complémentaires:**

**Numéro(s) magique(s) :**

Identique à [application/json](#) [\[JSON\]](#)

**Extension(s) de fichier :**

Identique à [application/json](#) [\[JSON\]](#)

**Code(s) de type de fichier Macintosh :**

Identique à [application/json](#) [\[JSON\]](#)

**Personne et adresse e-mail à contacter pour plus d'informations :**

Ian Hickson <ian@hixie.ch>

**Utilisation prévue :**

Commun

**Restrictions d'utilisation :**

Aucune restriction ne s'applique.

**Auteur:**

Ian Hickson <ian@hixie.ch>

**Changer de contrôleur :**

W3C

[Les fragments](#) utilisés avec [application/microdata+json](#) des ressources ont la même sémantique que lorsqu'ils sont utilisés avec [application/json](#) (à savoir, au moment de l'écriture, pas de sémantique du tout). [\[JSON\]](#)

## 17.6 [text/event-stream](#)

Cet enregistrement est destiné à l'examen de la communauté et sera soumis à l'IESG pour examen, approbation et enregistrement auprès de l'IANA.

**Tapez le nom :**

texte

**Nom du sous-type :**

flux d'événements

**Paramètres requis :**

Aucun paramètre

**Paramètres facultatifs :**

charset

Le `charset` paramètre peut être fourni. La valeur du paramètre doit être " `utf-8`". Ce paramètre ne sert à rien ; il n'est autorisé que pour la compatibilité avec les serveurs hérités.

**Considérations relatives à l'encodage :**

8 bits (toujours UTF-8)

**Considérations de sécurité :**

Un flux d'événements provenant d'une origine distincte de l'origine du contenu consommant le flux d'événements peut entraîner une fuite d'informations. Pour éviter cela, les agents utilisateurs doivent appliquer la sémantique CORS. [\[ALLER CHERCHER\]](#)

Les flux d'événements peuvent submerger un agent utilisateur ; un agent utilisateur est censé appliquer des restrictions appropriées pour éviter d'épuiser les ressources locales en raison d'une surabondance d'informations provenant d'un flux d'événements.

Les serveurs peuvent être débordés si une situation se développe dans laquelle le serveur oblige les clients à se reconnecter rapidement. Les serveurs doivent utiliser un code d'état 5xx pour indiquer les problèmes de capacité, car cela empêchera les clients conformes de se reconnecter automatiquement.

**Considérations d'interopérabilité :**

Les règles de traitement du contenu conforme et non conforme sont définies dans la présente spécification.

**Spécification publiée :**

Ce document est la spécification pertinente.

**Applications qui utilisent ce type de média :**

Navigateurs Web et outils utilisant des services Web.

**Informations Complémentaires:**

**Numéro(s) magique(s) :**

Aucune séquence d'octets ne peut identifier de manière unique un flux d'événements.

**Extension(s) de fichier :**

Aucune extension de fichier spécifique n'est recommandée pour ce type.

**Code(s) de type de fichier Macintosh :**

Aucun code de type de fichier Macintosh spécifique n'est recommandé pour ce type.

**Personne et adresse e-mail à contacter pour plus d'informations :**

Ian Hickson <ian@hixie.ch>

**Utilisation prévue :**

Commun

**Restrictions d'utilisation :**

Ce format ne devrait être utilisé que par des flux ouverts dynamiques servis à l'aide de HTTP ou d'un protocole similaire. Les ressources finies ne devraient pas être étiquetées avec ce type.

**Auteur:**

Ian Hickson <ian@hixie.ch>

**Changer de contrôleur :**

W3C

[Les fragments](#) n'ont aucun sens avec [text/event-stream](#) les ressources.

## 17.7 *web+* **préfixe de schéma**

Cette section décrit une convention à utiliser avec le registre de schéma d'URI IANA. Elle n'enregistre pas elle-même un régime spécifique. [\[RFC7595\]](#)

**Nom du schéma :**

Schémas commençant par les quatre caractères " *web+* " suivis d'une ou plusieurs lettres dans la plage *a-z*.

**Statut:**

Permanent

**Syntaxe du schéma :**

Spécifique au régime.

**Sémantique du schéma :**

Spécifique au régime.

**Considérations relatives à l'encodage :**

Tous *web+* les schémas " " doivent utiliser les encodages UTF-8, le cas échéant.

**Applications/protocoles qui utilisent ce nom de schéma :**

Spécifique au régime.

**Considérations d'interopérabilité :**

Le schéma devrait être utilisé dans le contexte d'applications Web.

**Considérations de sécurité :**

N'importe quelle page Web est capable d'enregistrer un gestionnaire pour tous *web+* les schémas " ". En tant que tels, ces schémas ne doivent pas être utilisés pour des fonctionnalités destinées à être des fonctionnalités de plateforme de base (par exemple, HTTP). De même, ces systèmes ne doivent pas stocker d'informations confidentielles dans leurs URL, telles que des noms

d'utilisateur, des mots de passe, des informations personnelles ou des noms de projet confidentiels.

**Contact:**

Ian Hickson <ian@hixie.ch>

**Changer de contrôleur :**

Ian Hickson <ian@hixie.ch>

**Les références:**

*Gestionnaires de schémas personnalisés* , HTML Living Standard : <https://html.spec.whatwg.org/#custom-handlers>

1. [Indice](#)
  1. [Éléments](#)
  2. [Catégories de contenu d'élément](#)
  3. [Les attributs](#)
  4. [Interfaces d'éléments](#)
  5. [Toutes les interfaces](#)
  6. [Événements](#)
  7. [En-têtes HTTP](#)
  8. [types MIME](#)

## Indice

Les sections suivantes ne couvrent que les éléments et fonctionnalités conformes.

## Éléments

*Cette section est non normative.*



# Liste des éléments

| Élément                        | Description                                                            | Catégories                                                                                                                          | Parents†                             | Enfants                               | Les attributs                                                                                                                                                                                                                                                                                                                  | Interface                                |
|--------------------------------|------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------|---------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------|
| <a href="#"><u>a</u></a>       | Lien hypertexte                                                        | <a href="#"><u>flux</u></a> ; <a href="#"><u>phrase</u></a> * ; <a href="#"><u>interactif</u></a> ; <a href="#"><u>palpable</u></a> | <a href="#"><u>formulation</u></a>   | <a href="#"><u>transparente</u></a> * | <a href="#"><u>globales</u></a> ; <a href="#"><u>href</u></a> ; <a href="#"><u>target</u></a> ; <a href="#"><u>download</u></a> ; <a href="#"><u>ping</u></a> ; <a href="#"><u>rel</u></a> ; <a href="#"><u>hreflang</u></a> ; <a href="#"><u>type</u></a> ; <a href="#"><u>referrerpolicy</u></a>                             | <a href="#"><u>HTMLAnchorElement</u></a> |
| <a href="#"><u>abbr</u></a>    | Abréviation                                                            | <a href="#"><u>flux</u></a> ; <a href="#"><u>phrase</u></a> ; <a href="#"><u>palpable</u></a>                                       | <a href="#"><u>formulation</u></a>   | <a href="#"><u>formulation</u></a>    | <a href="#"><u>globales</u></a>                                                                                                                                                                                                                                                                                                | <a href="#"><u>HTMLInputElement</u></a>  |
| <a href="#"><u>address</u></a> | Coordonnées d'une page ou d' <a href="#"><u>article</u></a> un élément | <a href="#"><u>flux</u></a> ; <a href="#"><u>palpable</u></a>                                                                       | <a href="#"><u>couleur</u></a>       | <a href="#"><u>flux</u></a> *         | <a href="#"><u>globales</u></a>                                                                                                                                                                                                                                                                                                | <a href="#"><u>HTMLInputElement</u></a>  |
| <a href="#"><u>area</u></a>    | Lien hypertexte ou zone morte sur une image cliq                       | <a href="#"><u>flux</u></a> ; <a href="#"><u>formulation</u></a>                                                                    | <a href="#"><u>formulation</u></a> * | vide                                  | <a href="#"><u>globales</u></a> ; <a href="#"><u>alt</u></a> ; <a href="#"><u>coords</u></a> ; <a href="#"><u>shape</u></a> ; <a href="#"><u>href</u></a> ; <a href="#"><u>target</u></a> ; <a href="#"><u>download</u></a> ; <a href="#"><u>ping</u></a> ; <a href="#"><u>rel</u></a> ; <a href="#"><u>referrerpolicy</u></a> | <a href="#"><u>HTMLAreaElement</u></a>   |

Liste des éléments

| Élément                                         | Description                                                 | Catégories                                                                      | Parents†                | Enfants                 | Les attributs            | Interface                                                            |
|-------------------------------------------------|-------------------------------------------------------------|---------------------------------------------------------------------------------|-------------------------|-------------------------|--------------------------|----------------------------------------------------------------------|
|                                                 | uable                                                       |                                                                                 |                         |                         |                          |                                                                      |
| <u>ar</u><br><u>ti</u><br><u>cl</u><br><u>e</u> | Composition autonome pouvant être syndiquée ou réutilisable | <a href="#">flux</a> ; <a href="#">sectionnement</a> ; <a href="#">palpable</a> | <a href="#">couleur</a> | <a href="#">couleur</a> | <a href="#">globales</a> | <a href="#">HTML</a><br><a href="#">LEl</a><br><a href="#">ement</a> |
| <u>as</u><br><u>id</u><br><u>e</u>              | Barre latérale pour le contenu lié de manière tangentielle  | <a href="#">flux</a> ; <a href="#">sectionnement</a> ; <a href="#">palpable</a> | <a href="#">couleur</a> | <a href="#">couleur</a> | <a href="#">globales</a> | <a href="#">HTML</a><br><a href="#">LEl</a><br><a href="#">ement</a> |

# Liste des éléments

| Élément               | Description                                                            | Catégories                                                                                                                        | Parents†                    | Enfants                                                                             | Les attributs                                                                                                                                                                                               | Interface                        |
|-----------------------|------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------|-----------------------------|-------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------|
| <a href="#">audio</a> | Lecteur audio                                                          | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">intégré</a> ; <a href="#">interactif</a> ; <a href="#">palpable</a> * | <a href="#">formulation</a> | <a href="#">source</a> * ; <a href="#">track</a> * ; <a href="#">transparente</a> * | <a href="#">globales</a> ; <a href="#">src</a> ; <a href="#">crossorigin</a> ; <a href="#">preload</a> ; <a href="#">autoplay</a> ; <a href="#">loop</a> ; <a href="#">muted</a> ; <a href="#">controls</a> | <a href="#">HTMLAudioElement</a> |
| <a href="#">b</a>     | Mots clés                                                              | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                                                          | <a href="#">formulation</a> | <a href="#">formulation</a>                                                         | <a href="#">globales</a>                                                                                                                                                                                    | <a href="#">HTMLElement</a>      |
| <a href="#">base</a>  | URL de base et cible par défaut pour les hyperliens et les formulaires | <a href="#">métadonnées</a>                                                                                                       | <a href="#">head</a>        | vide                                                                                | <a href="#">globales</a> ; <a href="#">href</a> ; <a href="#">target</a>                                                                                                                                    | <a href="#">HTMLBaseElement</a>  |
| <a href="#">bdi</a>   | Isolation de la direction                                              | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                                                          | <a href="#">formulation</a> | <a href="#">formulation</a>                                                         | <a href="#">globales</a>                                                                                                                                                                                    | <a href="#">HTMLElement</a>      |

# Liste des éléments

| Élément                                                       | Description                              | Catégories                                                               | Parents†                    | Enfants                     | Les attributs                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            | Interface        |
|---------------------------------------------------------------|------------------------------------------|--------------------------------------------------------------------------|-----------------------------|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------|
|                                                               | lité du texte                            |                                                                          |                             |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          |                  |
| <u>bd</u>                                                     | Formatage de la directionnalité du texte | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a> | <a href="#">formulation</a> | <a href="#">formulation</a> | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | HTMLElement      |
| <u>bl</u><br><u>oc</u><br><u>kg</u><br><u>uo</u><br><u>te</u> | Une section citée d'une autre source     | <a href="#">flux</a> ; <a href="#">palpable</a>                          | <a href="#">couler</a>      | <a href="#">couler</a>      | <a href="#">globales</a> ; <a href="#">cite</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | HTMLQuoteElement |
| <u>body</u>                                                   | Corps du document                        | aucun                                                                    | <a href="#">html</a>        | <a href="#">couler</a>      | <a href="#">globales</a> ; <a href="#">onafterprint</a> ; <a href="#">onbeforeprint</a> ; <a href="#">onbeforeunload</a> ; <a href="#">onhashchange</a> ; <a href="#">onlanguagechange</a> ; <a href="#">onmessage</a> ; <a href="#">onmessageerror</a> ; <a href="#">onoffline</a> ; <a href="#">ononline</a> ; <a href="#">onpagehide</a> ; <a href="#">onpageshow</a> ; <a href="#">onpopstate</a> ; <a href="#">onrejectionhandled</a> ; <a href="#">onstorage</a> ; <a href="#">onunhandledrejection</a> ; <a href="#">onunload</a> | HTMLBodyElement  |
| <u>br</u>                                                     | Saut de ligne, par exemple               | <a href="#">flux</a> ; <a href="#">formulation</a>                       | <a href="#">formulation</a> | vide                        | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | HTMLBRElement    |

# Liste des éléments

| Élément                 | Description                           | Catégories                                                                                                                                                                                                                        | Parents†                    | Enfants                       | Les attributs                                                                                                                                                                                                                                                                                                                                                                                                         | Interface                               |
|-------------------------|---------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|
|                         | edans un poème ou une adresse postale |                                                                                                                                                                                                                                   |                             |                               |                                                                                                                                                                                                                                                                                                                                                                                                                       |                                         |
| <a href="#">boutton</a> | Bouton de contrôle                    | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">interactif</a> ; <a href="#">listé</a> ; <a href="#">étiquetable</a> ; <a href="#">soumissable</a> ; <a href="#">associé au formulaire</a> ; <a href="#">palpable</a> | <a href="#">formulation</a> | <a href="#">formulation</a> * | <a href="#">globales</a> ; <a href="#">disabled</a> ; <a href="#">form</a> ; <a href="#">formaction</a> ; <a href="#">formmenctype</a> ; <a href="#">formmethod</a> ; <a href="#">formnovalidate</a> ; <a href="#">formtarget</a> ; <a href="#">name</a> ; <a href="#">popoverhidetarget</a> ; <a href="#">popovershowtarget</a> ; <a href="#">popovertoggletarget</a> ; <a href="#">type</a> ; <a href="#">value</a> | <a href="#">HTMLButtonElement</a>       |
| <a href="#">cas</a>     | Casnevas bitmap inscriptible          | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">intégré</a> ; <a href="#">palpable</a>                                                                                                                                | <a href="#">formulation</a> | <a href="#">transparent</a>   | <a href="#">globales</a> ; <a href="#">width</a> ; <a href="#">height</a>                                                                                                                                                                                                                                                                                                                                             | <a href="#">HTMLCanvasElement</a>       |
| <a href="#">caption</a> | Légende du tableau                    | aucun                                                                                                                                                                                                                             | <a href="#">table</a>       | <a href="#">flux</a> *        | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                              | <a href="#">HTMLTableCaptionElement</a> |
| <a href="#">cite</a>    | Titre d'une                           | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">palpable</a>                                                                                                                                                          | <a href="#">formulation</a> | <a href="#">formulation</a>   | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                              | <a href="#">HTMLElement</a>             |

# Liste des éléments

| Élément          | Description                       | Catégories                                    | Parents†           | Enfants                                                      | Les attributs                  | Interface                          |
|------------------|-----------------------------------|-----------------------------------------------|--------------------|--------------------------------------------------------------|--------------------------------|------------------------------------|
|                  | oeuvre                            |                                               |                    |                                                              |                                |                                    |
| <u>code</u>      | Code informatique                 | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u> | <u>formulation</u> | <u>formulation</u>                                           | <u>globales</u>                | HTML<br>Element                    |
| <u>col</u>       | Colonne du tableau                | aucun                                         | <u>colgroup</u>    | vide                                                         | <u>globales</u> ; <u>span</u>  | HTML<br>Table<br>Column<br>Element |
| <u>colgroup</u>  | Groupe de colonnes dans une table | aucun                                         | <u>table</u>       | <u>col*</u> ; <u>template*</u>                               | <u>globales</u> ; <u>span</u>  | HTML<br>Table<br>Column<br>Element |
| <u>data</u>      | Équivalent lisible par machine    | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u> | <u>formulation</u> | <u>formulation</u>                                           | <u>globales</u> ; <u>value</u> | HTML<br>Data<br>Element            |
| <u>data list</u> | Conteneur d'option                | <u>flux</u> ; <u>formulation</u>              | <u>formulation</u> | <u>phrasé*</u> ; <u>option*</u> ; <u>éléments de support</u> | <u>globales</u>                | HTML<br>Data<br>List<br>Element    |

# Liste des éléments

| Élément                    | Description                                     | Catégories                                                                   | Parents†                                   | Enfants                                           | Les attributs                                                              | Interface                          |
|----------------------------|-------------------------------------------------|------------------------------------------------------------------------------|--------------------------------------------|---------------------------------------------------|----------------------------------------------------------------------------|------------------------------------|
|                            | ns pour le contrôle de zone de liste déroulante |                                                                              |                                            | <a href="#">de script</a> *                       |                                                                            |                                    |
| <a href="#">dd</a>         | Contenu des éléments correspondants             | aucun                                                                        | <a href="#">dl</a> ; <a href="#">div</a> * | <a href="#">couler</a>                            | <a href="#">globales</a>                                                   | <a href="#">HTML Element</a>       |
| <a href="#">de l</a>       | Une suppression du document                     | <a href="#">flux</a> ; <a href="#">phrase</a> *; <a href="#">palpable</a>    | <a href="#">formulation</a>                | <a href="#">transparent</a>                       | <a href="#">globales</a> ; <a href="#">cite</a> ; <a href="#">datetime</a> | <a href="#">HTML Model Element</a> |
| <a href="#">de ta il s</a> | Contrôle                                        | <a href="#">flux</a> ; <a href="#">interactif</a> ; <a href="#">palpable</a> | <a href="#">couler</a>                     | <a href="#">summary</a> *; <a href="#">couler</a> | <a href="#">globales</a> ; <a href="#">open</a>                            | <a href="#">HTML Details</a>       |

Liste des éléments

| Élément                | Description                                | Catégories                                                               | Parents†                                  | Enfants                       | Les attributs                                   | Interface                          |
|------------------------|--------------------------------------------|--------------------------------------------------------------------------|-------------------------------------------|-------------------------------|-------------------------------------------------|------------------------------------|
|                        | de la divulgation pour masquer les détails |                                                                          |                                           |                               |                                                 | <a href="#">lement</a>             |
| <a href="#">dfn</a>    | Définition de l'instance                   | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a> | <a href="#">formulation</a>               | <a href="#">formulation</a> * | <a href="#">globales</a>                        | <a href="#">HTML Element</a>       |
| <a href="#">dialog</a> | Boîte de dialogue ou fenêtre               | <a href="#">couler</a>                                                   | <a href="#">couler</a>                    | <a href="#">couler</a>        | <a href="#">globales</a> ; <a href="#">open</a> | <a href="#">HTML DialogElement</a> |
| <a href="#">div</a>    | Conteneur de flux générique ou con         | <a href="#">flux</a> ; <a href="#">palpable</a>                          | <a href="#">flux</a> ; <a href="#">dl</a> | <a href="#">couler</a>        | <a href="#">globales</a>                        | <a href="#">HTML DivElement</a>    |



Liste des éléments

| Élément   | Description                                                               | Catégories                    | Parents†                 | Enfants                                                                         | Les attributs   | Interface                                                                      |
|-----------|---------------------------------------------------------------------------|-------------------------------|--------------------------|---------------------------------------------------------------------------------|-----------------|--------------------------------------------------------------------------------|
|           | teneur pour les groupes nom-va leur dans s <u>d</u> <u>l</u> e s éléments |                               |                          |                                                                                 |                 |                                                                                |
| <u>d1</u> | Liste d'associations composée de zéro ou plusieurs groupes nom-           | <u>flux</u> ; <u>palpable</u> | <u>coul</u><br><u>er</u> | <u>dt</u> * ; <u>dd</u> * ; <u>div</u> * ; <u>éléments de support de script</u> | <u>globales</u> | <u>HTM</u><br><u>LDL</u><br><u>ist</u><br><u>Ele</u><br><u>men</u><br><u>t</u> |

# Liste des éléments

| Élément           | Description                         | Catégories                                                                         | Parents†                 | Enfants                          | Les attributs                                                                               | Interface              |
|-------------------|-------------------------------------|------------------------------------------------------------------------------------|--------------------------|----------------------------------|---------------------------------------------------------------------------------------------|------------------------|
|                   | val eur                             |                                                                                    |                          |                                  |                                                                                             |                        |
| <u>dt</u>         | Légende des éléments correspondants | aucun                                                                              | <u>dl</u> ; <u>div</u> * | <u>flux</u> *                    | <u>globales</u>                                                                             | HTML Element           |
| <u>em</u>         | Accent mis sur le stress            | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u>                                      | <u>formulation</u>       | <u>formulation</u>               | <u>globales</u>                                                                             | HTML Element           |
| <u>embed</u>      | Brancher                            | <u>flux</u> ; <u>phrasé</u> ; <u>intégré</u> ; <u>interactif</u> ; <u>palpable</u> | <u>formulation</u>       | vide                             | <u>globales</u> ; <u>src</u> ; <u>type</u> ; <u>width</u> ; <u>height</u> ; n'importe quel* | HTML Embed Element     |
| <u>fieldset</u>   | Groupe de champs de formulaire      | <u>flux</u> ; <u>listé</u> ; <u>associé au formulaire</u> ; <u>palpable</u>        | <u>couleur</u>           | <u>legend</u> * ; <u>couleur</u> | <u>globales</u> ; <u>disabled</u> ; <u>form</u> ; <u>name</u>                               | HTML Field Set Element |
| <u>figcaption</u> | Légende pour                        | aucun                                                                              | <u>figure</u>            | <u>couleur</u>                   | <u>globales</u>                                                                             | HTML Element           |

Liste des éléments

| Élément                                                                        | Description                              | Catégories                                    | Parents†                                                    | Enfants                                                   | Les attributs                                                                                                                                                                            | Interface                              |
|--------------------------------------------------------------------------------|------------------------------------------|-----------------------------------------------|-------------------------------------------------------------|-----------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------|
|                                                                                | <u>rfi</u><br><u>gur</u><br><u>e</u>     |                                               |                                                             |                                                           |                                                                                                                                                                                          |                                        |
| <u>fi</u><br><u>gu</u><br><u>re</u>                                            | Figure avec légende facultative          | <u>flux</u> ; <u>palpable</u>                 | <u>coul</u><br><u>er</u>                                    | <u>figcapt</u><br><u>ion</u> * ; <u>co</u><br><u>uler</u> | <u>globales</u>                                                                                                                                                                          | HTM<br>LEl<br>eme<br>nt                |
| <u>fo</u><br><u>ot</u><br><u>er</u>                                            | Pied de page d'une page ou d'une section | <u>flux</u> ; <u>palpable</u>                 | <u>coul</u><br><u>er</u>                                    | <u>flux</u> *                                             | <u>globales</u>                                                                                                                                                                          | HTM<br>LEl<br>eme<br>nt                |
| <u>fo</u><br><u>rm</u>                                                         | Formulaire soumis par l'utilisateur      | <u>flux</u> ; <u>palpable</u>                 | <u>coul</u><br><u>er</u>                                    | <u>flux</u> *                                             | <u>globales</u> ; <u>accept-</u><br><u>charset</u> ; <u>action</u> ; <u>autocomplete</u> ; <u>enctype</u> ; <u>method</u> ; <u>name</u> ; <u>novalidate</u> ; <u>rel</u> ; <u>target</u> | HTM<br>LFo<br>rmE<br>lem<br>ent        |
| <u>h1</u><br>, <u>h</u><br><u>2</u> ,<br><u>h3</u><br>, <u>h</u><br><u>4</u> , | Titre                                    | <u>flux</u> ; <u>cap</u> ;<br><u>palpable</u> | <u>legende</u> ; <u>somma</u><br><u>ry</u> ; <u>couleur</u> | <u>formulation</u>                                        | <u>globales</u>                                                                                                                                                                          | HTM<br>LHe<br>adi<br>ngE<br>lem<br>ent |

Liste des éléments

| Élément                                 | Description                                                        | Catégories                                      | Parents†                                                                   | Enfants                                                                                                                                                | Les attributs            | Interface                        |
|-----------------------------------------|--------------------------------------------------------------------|-------------------------------------------------|----------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------|----------------------------------|
| <a href="#">h5</a> , <a href="#">h6</a> |                                                                    |                                                 |                                                                            |                                                                                                                                                        |                          |                                  |
| <a href="#">head</a>                    | Conteneur pour les métadonnées de document                         | aucun                                           | <a href="#">html</a>                                                       | <a href="#">contenu des métadonnées</a> *                                                                                                              | <a href="#">globales</a> | <a href="#">HTMLHeadElement</a>  |
| <a href="#">header</a>                  | Aides d'introduction ou de navigation pour une page ou une section | <a href="#">flux</a> ; <a href="#">palpable</a> | <a href="#">couleur</a>                                                    | <a href="#">flux</a> *                                                                                                                                 | <a href="#">globales</a> | <a href="#">HTMLInputElement</a> |
| <a href="#">highlight</a>               | Conteneur                                                          | <a href="#">flux</a> ; <a href="#">palpable</a> | <a href="#">legend</a> ; <a href="#">summary</a> ; <a href="#">caption</a> | <a href="#">h1</a> ; <a href="#">h2</a> ; <a href="#">h3</a> ; <a href="#">h4</a> ; <a href="#">h5</a> ; <a href="#">h6</a> ; <a href="#">éléments</a> | <a href="#">globales</a> | <a href="#">HTMLInputElement</a> |

# Liste des éléments

| Élément                | Description                                  | Catégories                                                                                                                                                                                                                                                              | Parents†                                         | Enfants                                         | Les attributs                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | Interface                                                  |
|------------------------|----------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------|-------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|
|                        | d'en-tête                                    |                                                                                                                                                                                                                                                                         | <a href="#">ouler</a>                            | <a href="#">de support de script</a>            |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 |                                                            |
| <a href="#">hr</a>     | Pa<br>use<br>thé<br>mat<br>iqu<br>e          | <a href="#">couler</a>                                                                                                                                                                                                                                                  | <a href="#">coul<br/>er</a>                      | vide                                            | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | <a href="#">HTM<br/>LHR<br/>Ele<br/>men<br/>t</a>          |
| <a href="#">html</a>   | Élé<br>me<br>nt<br>raci<br>ne                | aucun                                                                                                                                                                                                                                                                   | aucun*                                           | <a href="#">head</a> * ; <a href="#">body</a> * | <a href="#">globales</a> ; <a href="#">manifest</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                             | <a href="#">HTM<br/>LHt<br/>mLE<br/>lem<br/>ent</a>        |
| <a href="#">i</a>      | Voi<br>x<br>alte<br>rna<br>tive              | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                                                                                                                                                                                                | <a href="#">form<br/>ulati<br/>on</a>            | <a href="#">formulati<br/>on</a>                | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                        | <a href="#">HTM<br/>LEl<br/>eme<br/>nt</a>                 |
| <a href="#">iframe</a> | Enf<br>ant<br>nav<br>iga<br>ble              | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">intégré</a> ; <a href="#">interactif</a> ; <a href="#">palpable</a>                                                                                                                                         | <a href="#">form<br/>ulati<br/>on</a>            | vide                                            | <a href="#">globales</a> ; <a href="#">src</a> ; <a href="#">srcdoc</a> ; <a href="#">name</a> ; <a href="#">sandbox</a> ; <a href="#">alloww</a> ; <a href="#">allowfullscreen</a> ; <a href="#">width</a> ; <a href="#">height</a> ; <a href="#">referrerpolicy</a> ; <a href="#">loading</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                 | <a href="#">HTM<br/>LIF<br/>ram<br/>eEl<br/>eme<br/>nt</a> |
| <a href="#">img</a>    | Ima<br>ge                                    | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">intégré</a> ; <a href="#">interactif</a> * ; <a href="#">associé au formulaire</a> ; <a href="#">palpable</a>                                                                                               | <a href="#">phrasé</a> ; <a href="#">picture</a> | vide                                            | <a href="#">globales</a> ; <a href="#">alt</a> ; <a href="#">src</a> ; <a href="#">srcset</a> ; <a href="#">sizes</a> ; <a href="#">crossorigin</a> ; <a href="#">usemap</a> ; <a href="#">ismap</a> ; <a href="#">width</a> ; <a href="#">height</a> ; <a href="#">referrerpolicy</a> ; <a href="#">decoding</a> ; <a href="#">loading</a> ; <a href="#">fetchpriority</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | <a href="#">HTM<br/>LIm<br/>age<br/>Ele<br/>men<br/>t</a>  |
| <a href="#">input</a>  | Co<br>ntr<br>ôle<br>de<br>for<br>mul<br>aire | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">interactif</a> * ; <a href="#">listé</a> ; <a href="#">étiquetable</a> ; <a href="#">soumissable</a> ; <a href="#">réinitialisable</a> ; <a href="#">associé au formulaire</a> ; <a href="#">palpable</a> * | <a href="#">form<br/>ulati<br/>on</a>            | vide                                            | <a href="#">globales</a> ; <a href="#">accept</a> ; <a href="#">alt</a> ; <a href="#">autocomplete</a> ; <a href="#">checked</a> ; <a href="#">dirname</a> ; <a href="#">disabled</a> ; <a href="#">form</a> ; <a href="#">formaction</a> ; <a href="#">formenctype</a> ; <a href="#">formmethod</a> ; <a href="#">formnovalidate</a> ; <a href="#">formtarget</a> ; <a href="#">height</a> ; <a href="#">list</a> ; <a href="#">max</a> ; <a href="#">maxlength</a> ; <a href="#">min</a> ; <a href="#">minlength</a> ; <a href="#">multiple</a> ; <a href="#">name</a> ; <a href="#">pattern</a> ; <a href="#">placeholder</a> ; <a href="#">popoverhidetarget</a> ; <a href="#">popovershowtarget</a> ; <a href="#">popovertoggletarget</a> ; <a href="#">readonly</a> ; <a href="#">required</a> ; <a href="#">size</a> ; <a href="#">src</a> ; <a href="#">step</a> ; <a href="#">type</a> ; <a href="#">value</a> ; <a href="#">width</a> | <a href="#">HTM<br/>LIn<br/>put<br/>Ele<br/>men<br/>t</a>  |

# Liste des éléments

| Élément                | Description                           | Catégories                                                                                            | Parents†                                                             | Enfants                                                   | Les attributs                                                                                                                                                                                                                                                                                                                                                                                                                                                    | Interface         |
|------------------------|---------------------------------------|-------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------|-----------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------|
| <a href="#">ins</a>    | Un complément au document             | <a href="#">flux</a> ; <a href="#">phrasé*</a> ; <a href="#">palpable</a>                             | <a href="#">formulation</a>                                          | <a href="#">transparent</a>                               | <a href="#">globales</a> ; <a href="#">cite</a> ; <a href="#">datetime</a>                                                                                                                                                                                                                                                                                                                                                                                       | HTMLModelElement  |
| <a href="#">kbd</a>    | Entrée utilisateur                    | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                              | <a href="#">formulation</a>                                          | <a href="#">formulation</a>                               | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                                                                         | HTMLInputElement  |
| <a href="#">label</a>  | Légende d'un champ de formulaire      | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">interactif</a> ; <a href="#">palpable</a> | <a href="#">formulation</a>                                          | <a href="#">formulation*</a>                              | <a href="#">globales</a> ; <a href="#">for</a>                                                                                                                                                                                                                                                                                                                                                                                                                   | HTMLLabelElement  |
| <a href="#">legend</a> | Légende pour <a href="#">fieldset</a> | aucun                                                                                                 | <a href="#">fieldset</a>                                             | <a href="#">phrasé</a> ; <a href="#">contenu du titre</a> | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                                                                                                                         | HTMLLegendElement |
| <a href="#">li</a>     | Élément de liste                      | aucun                                                                                                 | <a href="#">ol</a> ; <a href="#">ul</a> ; <a href="#">menu*</a>      | <a href="#">couleur</a>                                   | <a href="#">globales</a> ; <a href="#">value*</a>                                                                                                                                                                                                                                                                                                                                                                                                                | HTMLLIElement     |
| <a href="#">link</a>   | Métadonnées                           | <a href="#">métadonnées</a> ; <a href="#">flux*</a> ; <a href="#">formulation*</a>                    | <a href="#">head</a> ; <a href="#">noscript*</a> ; <a href="#">f</a> | vide                                                      | <a href="#">globales</a> ; <a href="#">href</a> ; <a href="#">crossorigin</a> ; <a href="#">rel</a> ; <a href="#">as</a> ; <a href="#">media</a> ; <a href="#">hreflang</a> ; <a href="#">type</a> ; <a href="#">sizes</a> ; <a href="#">imagesrcset</a> ; <a href="#">imagesizes</a> ; <a href="#">referrerpolicy</a> ; <a href="#">integrity</a> ; <a href="#">blocking</a> ; <a href="#">color</a> ; <a href="#">disabled</a> ; <a href="#">fetchpriority</a> | HTMLLinkElement   |

Liste des éléments

| Élément                | Description                                    | Catégories                                                                                         | Parents†                      | Enfants                                                            | Les attributs                                   | Interface                         |
|------------------------|------------------------------------------------|----------------------------------------------------------------------------------------------------|-------------------------------|--------------------------------------------------------------------|-------------------------------------------------|-----------------------------------|
|                        | du lien                                        |                                                                                                    | <a href="#">formulation</a> * |                                                                    |                                                 |                                   |
| <a href="#">main</a>   | Conteneur pour le contenu dominant du document | <a href="#">flux</a> ; <a href="#">palpable</a>                                                    | <a href="#">flux</a> *        | <a href="#">couler</a>                                             | <a href="#">globales</a>                        | <a href="#">HTML Element</a>      |
| <a href="#">map</a>    | <a href="#">Carte-image</a>                    | <a href="#">flux</a> ; <a href="#">phrase</a> * ; <a href="#">palpable</a>                         | <a href="#">formulation</a>   | <a href="#">transparente</a> ; <a href="#">area</a> *              | <a href="#">globales</a> ; <a href="#">name</a> | <a href="#">HTML Map Element</a>  |
| <a href="#">mark</a>   | Souligner                                      | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">palpable</a>                           | <a href="#">formulation</a>   | <a href="#">formulation</a>                                        | <a href="#">globales</a>                        | <a href="#">HTML Element</a>      |
| <a href="#">MathML</a> | Racine MathML                                  | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">intégré</a> ; <a href="#">palpable</a> | <a href="#">formulation</a>   | par <a href="#">[MATHML]</a>                                       | par <a href="#">[MATHML]</a>                    | <a href="#">Element</a>           |
| <a href="#">menu</a>   | Menu de commandes                              | <a href="#">flux</a> ; <a href="#">palpable</a> *                                                  | <a href="#">couler</a>        | <a href="#">li</a> ; <a href="#">éléments de support de script</a> | <a href="#">globales</a>                        | <a href="#">HTML Menu Element</a> |

# Liste des éléments

| Élément                  | Description                       | Catégories                                                                                                                                                                                        | Parents†                                                                          | Enfants                       | Les attributs                                                                                                                                                         | Interface                          |
|--------------------------|-----------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------|-------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------|
| <a href="#">meta</a>     | Métadonnées textuelles            | <a href="#">métadonnées</a> ; <a href="#">flux</a> * ; <a href="#">formulation</a> *                                                                                                              | <a href="#">head</a> ; <a href="#">noscript</a> * ; <a href="#">formulation</a> * | vide                          | <a href="#">globales</a> ; <a href="#">name</a> ; <a href="#">http-equiv</a> ; <a href="#">content</a> ; <a href="#">charset</a> ; <a href="#">media</a>              | <a href="#">HTMLMetaElement</a>    |
| <a href="#">meter</a>    | Jauge                             | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">étiquetable</a> ; <a href="#">palpable</a>                                                                                            | <a href="#">formulation</a>                                                       | <a href="#">formulation</a> * | <a href="#">globales</a> ; <a href="#">value</a> ; <a href="#">min</a> ; <a href="#">max</a> ; <a href="#">low</a> ; <a href="#">high</a> ; <a href="#">optimum</a>   | <a href="#">HTMLMeterElement</a>   |
| <a href="#">nav</a>      | Section avec liens de navigation  | <a href="#">flux</a> ; <a href="#">sectionnement</a> ; <a href="#">palpable</a>                                                                                                                   | <a href="#">couleur</a>                                                           | <a href="#">couleur</a>       | <a href="#">globales</a>                                                                                                                                              | <a href="#">HTMLSectionElement</a> |
| <a href="#">noscript</a> | Contenu de secours pour le script | <a href="#">métadonnées</a> ; <a href="#">flux</a> ; <a href="#">formulation</a>                                                                                                                  | <a href="#">head</a> * ; <a href="#">formulation</a> *                            | varie*                        | <a href="#">globales</a>                                                                                                                                              | <a href="#">HTMLSectionElement</a> |
| <a href="#">object</a>   | Image, enfant navigable ou        | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">intégré</a> ; <a href="#">interactif</a> * ; <a href="#">listé</a> ; <a href="#">associé au formulaire</a> ; <a href="#">palpable</a> | <a href="#">formulation</a>                                                       | <a href="#">transparent</a>   | <a href="#">globales</a> ; <a href="#">data</a> ; <a href="#">type</a> ; <a href="#">name</a> ; <a href="#">form</a> ; <a href="#">width</a> ; <a href="#">height</a> | <a href="#">HTMLObjectElement</a>  |



# Liste des éléments

| Élément                  | Description                             | Catégories                                                                                                                                                                                               | Parents†                                                                     | Enfants                                                                | Les attributs                                                                                                                  | Interface                                                                                         |
|--------------------------|-----------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------|------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------|
|                          | <a href="#">plugg-in</a>                |                                                                                                                                                                                                          |                                                                              |                                                                        |                                                                                                                                |                                                                                                   |
| <a href="#">ol</a>       | Liste ordonnée                          | <a href="#">flux</a> ; <a href="#">palpable</a> *                                                                                                                                                        | <a href="#">couleur</a>                                                      | <a href="#">li</a> ; <a href="#">éléments de support de script</a>     | <a href="#">globales</a> ; <a href="#">reversed</a> ; <a href="#">start</a> ; <a href="#">type</a>                             | <a href="#">HTML</a><br><a href="#">LOList</a><br><a href="#">Element</a><br><a href="#">t</a>    |
| <a href="#">optgroup</a> | Groupe d'options dans une zone de liste | aucun                                                                                                                                                                                                    | <a href="#">select</a>                                                       | <a href="#">option</a> ; <a href="#">éléments de support de script</a> | <a href="#">globales</a> ; <a href="#">disabled</a> ; <a href="#">label</a>                                                    | <a href="#">HTML</a><br><a href="#">LOptGroup</a><br><a href="#">Element</a><br><a href="#">t</a> |
| <a href="#">option</a>   | Option dans un champ liste ou combo     | aucun                                                                                                                                                                                                    | <a href="#">select</a> ; <a href="#">datalist</a> ; <a href="#">optgroup</a> | <a href="#">texte</a> *                                                | <a href="#">globales</a> ; <a href="#">disabled</a> ; <a href="#">label</a> ; <a href="#">selected</a> ; <a href="#">value</a> | <a href="#">HTML</a><br><a href="#">LOption</a><br><a href="#">Element</a><br><a href="#">nt</a>  |
| <a href="#">output</a>   | Valeur de sortie calculée               | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">listé</a> ; <a href="#">étiquetable</a> ; <a href="#">réinitialisable</a> ; <a href="#">associé au formulaire</a> ; <a href="#">palpable</a> | <a href="#">formulation</a>                                                  | <a href="#">formulation</a>                                            | <a href="#">globales</a> ; <a href="#">for</a> ; <a href="#">form</a> ; <a href="#">name</a>                                   | <a href="#">HTML</a><br><a href="#">LOutput</a><br><a href="#">Element</a><br><a href="#">nt</a>  |

# Liste des éléments

| Élément                  | Description                  | Catégories                                                                                              | Parents†                    | Enfants                                                                                          | Les attributs                                                          | Interface                            |
|--------------------------|------------------------------|---------------------------------------------------------------------------------------------------------|-----------------------------|--------------------------------------------------------------------------------------------------|------------------------------------------------------------------------|--------------------------------------|
| <a href="#">p</a>        | Paragraphe                   | <a href="#">flux</a> ; <a href="#">palpable</a>                                                         | <a href="#">couleur</a>     | <a href="#">formulation</a>                                                                      | <a href="#">globales</a>                                               | <a href="#">HTMLParagraphElement</a> |
| <a href="#">picture</a>  | Image                        | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">intégrés</a> ; <a href="#">palpable</a>     | <a href="#">formulation</a> | <a href="#">source*</a> ; un <a href="#">img</a> ; <a href="#">éléments de support de script</a> | <a href="#">globales</a>                                               | <a href="#">HTMLPictureElement</a>   |
| <a href="#">pre</a>      | Bloc de texte préformaté     | <a href="#">flux</a> ; <a href="#">palpable</a>                                                         | <a href="#">couleur</a>     | <a href="#">formulation</a>                                                                      | <a href="#">globales</a>                                               | <a href="#">HTMLPreElement</a>       |
| <a href="#">progress</a> | Barre de progression         | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">étiquetables</a> ; <a href="#">palpable</a> | <a href="#">formulation</a> | <a href="#">formulation</a> *                                                                    | <a href="#">globales</a> ; <a href="#">value</a> ; <a href="#">max</a> | <a href="#">HTMLProgressElement</a>  |
| <a href="#">q</a>        | Citation                     | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                                | <a href="#">formulation</a> | <a href="#">formulation</a>                                                                      | <a href="#">globales</a> ; <a href="#">cite</a>                        | <a href="#">HTMLQuoteElement</a>     |
| <a href="#">rp</a>       | Parent hès pour le texte d'a | aucun                                                                                                   | <a href="#">ruby</a>        | <a href="#">texte</a>                                                                            | <a href="#">globales</a>                                               | <a href="#">HTMLLElement</a>         |

# Liste des éléments

| Élément       | Description             | Catégories                                                                              | Parents†                                                           | Enfants                                     | Les attributs                                                                                                                                                                                       | Interface                                                             |
|---------------|-------------------------|-----------------------------------------------------------------------------------------|--------------------------------------------------------------------|---------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------|
|               | Annotation Ruby         |                                                                                         |                                                                    |                                             |                                                                                                                                                                                                     |                                                                       |
| <u>rt</u>     | Texte d'annotation Ruby | aucun                                                                                   | <u>ruby</u>                                                        | <u>formulation</u>                          | <u>globales</u>                                                                                                                                                                                     | <u>HTML</u><br><u>LEl</u><br><u>ement</u>                             |
| <u>ruby</u>   | Annotation(s) Ruby      | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u>                                           | <u>formulation</u>                                                 | <u>phrasé</u> ; <u>rt</u> ; <u>rp</u> *     | <u>globales</u>                                                                                                                                                                                     | <u>HTML</u><br><u>LEl</u><br><u>ement</u>                             |
| <u>s</u>      | Texte inexact           | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u>                                           | <u>formulation</u>                                                 | <u>formulation</u>                          | <u>globales</u>                                                                                                                                                                                     | <u>HTML</u><br><u>LEl</u><br><u>ement</u>                             |
| <u>samp</u>   | Sorte informatique      | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u>                                           | <u>formulation</u>                                                 | <u>formulation</u>                          | <u>globales</u>                                                                                                                                                                                     | <u>HTML</u><br><u>LEl</u><br><u>ement</u>                             |
| <u>script</u> | Script intégré          | <u>métadonnées</u> ; <u>flux</u> ; <u>phrasé</u> ; <u>prenant en charge les scripts</u> | <u>head</u> ; <u>phrasé</u> ; <u>prenant en charge les scripts</u> | script, données ou documentation de script* | <u>globales</u> ; <u>src</u> ; <u>type</u> ; <u>nomodule</u> ; <u>async</u> ; <u>defer</u> ; <u>crossorigin</u> ; <u>integrity</u> ; <u>referrerpolicy</u> ; <u>blocking</u> ; <u>fetchpriority</u> | <u>HTML</u><br><u>LSc</u><br><u>ript</u><br><u>El</u><br><u>ement</u> |

# Liste des éléments

| Élément                 | Description                                 | Catégories                                                                                                                                                                                                                                                          | Parents†                    | Enfants                                                                                           | Les attributs                                                                                                                                                                                                 | Interface                           |
|-------------------------|---------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------|---------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------|
| <a href="#">section</a> | Document générique ou section d'application | <a href="#">flux</a> ; <a href="#">sectionnement</a> ; <a href="#">palpable</a>                                                                                                                                                                                     | <a href="#">couler</a>      | <a href="#">couler</a>                                                                            | <a href="#">globales</a>                                                                                                                                                                                      | <a href="#">HTML Element</a>        |
| <a href="#">select</a>  | Contrôle de zone de liste                   | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">interactif</a> ; <a href="#">listé</a> ; <a href="#">étiquetable</a> ; <a href="#">soumissable</a> ; <a href="#">réinitialisable</a> ; <a href="#">associé au formulaire</a> ; <a href="#">palpable</a> | <a href="#">formulation</a> | <a href="#">option</a> ; <a href="#">optgroup</a> ; <a href="#">éléments de support de script</a> | <a href="#">globales</a> ; <a href="#">autocomplete</a> ; <a href="#">disabled</a> ; <a href="#">form</a> ; <a href="#">multiple</a> ; <a href="#">name</a> ; <a href="#">required</a> ; <a href="#">size</a> | <a href="#">HTML Select Element</a> |
| <a href="#">slot</a>    | Élément d'arbre d'ombre                     | <a href="#">flux</a> ; <a href="#">formulation</a>                                                                                                                                                                                                                  | <a href="#">formulation</a> | <a href="#">transparent</a>                                                                       | <a href="#">globales</a> ; <a href="#">name</a>                                                                                                                                                               | <a href="#">HTML Slot Element</a>   |
| <a href="#">summary</a> | Commentaire latéral                         | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">palpable</a>                                                                                                                                                                                            | <a href="#">formulation</a> | <a href="#">formulation</a>                                                                       | <a href="#">globales</a>                                                                                                                                                                                      | <a href="#">HTML Element</a>        |

# Liste des éléments

| Élément       | Description                                                                                                                                                                             | Catégories                                    | Parents†                                           | Enfants            | Les attributs                                                                                                              | Interface                  |
|---------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------|----------------------------------------------------|--------------------|----------------------------------------------------------------------------------------------------------------------------|----------------------------|
| <u>source</u> | Sourc<br>e<br>d'i<br>ma<br>ge<br>pou<br>r <u>im</u><br><u>g</u> ou<br>sou<br>rce<br>mul<br>tim<br>édi<br>a<br>pou<br>r <u>vi</u><br><u>deo</u><br>ou <u>a</u><br><u>udi</u><br><u>o</u> | aucun                                         | <u>picture</u> ;<br><u>video</u> ;<br><u>audio</u> | vide               | <u>globales</u> ; <u>src</u> ; <u>type</u> ; <u>srcset</u> ; <u>sizes</u> ; <u>media</u> ;<br><u>width</u> ; <u>height</u> | HTML<br>LSource<br>Element |
| <u>span</u>   | Conte<br>neur<br>de<br>phr<br>asé<br>gén<br>ériq<br>ue                                                                                                                                  | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u> | <u>formulation</u>                                 | <u>formulation</u> | <u>globales</u>                                                                                                            | HTML<br>LSpan<br>Element   |
| <u>strong</u> | Imp<br>ort<br>anc<br>e                                                                                                                                                                  | <u>flux</u> ; <u>phrasé</u> ; <u>palpable</u> | <u>formulation</u>                                 | <u>formulation</u> | <u>globales</u>                                                                                                            | HTML<br>LElement           |
| <u>style</u>  | Info<br>rm<br>atio<br>ns<br>de<br>styl<br>e<br>inté                                                                                                                                     | <u>métadonné</u><br><u>es</u>                 | <u>head</u> ;<br><u>noscript</u><br><u>t*</u>      | texte*             | <u>globales</u> ; <u>media</u> ; <u>blocking</u>                                                                           | HTML<br>LStyle<br>Element  |

# Liste des éléments

| Élément                                    | Description                      | Catégories                                                                                         | Parents†                    | Enfants                                                                                                                                                                                               | Les attributs             | Interface                                                                                                                  |
|--------------------------------------------|----------------------------------|----------------------------------------------------------------------------------------------------|-----------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------|----------------------------------------------------------------------------------------------------------------------------|
|                                            | grés                             |                                                                                                    |                             |                                                                                                                                                                                                       |                           |                                                                                                                            |
| <a href="#">sub</a>                        | Indice                           | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                           | <a href="#">formulation</a> | <a href="#">formulation</a>                                                                                                                                                                           | <a href="#">globales</a>  | <a href="#">HTML</a><br><a href="#">LEl</a><br><a href="#">ement</a><br><a href="#">nt</a>                                 |
| <a href="#">summary</a>                    | Légende pour de tableaux         | aucun                                                                                              | <a href="#">détails</a>     | <a href="#">phrasé</a> ; <a href="#">contenu du titre</a>                                                                                                                                             | <a href="#">globales</a>  | <a href="#">HTML</a><br><a href="#">LEl</a><br><a href="#">ement</a><br><a href="#">nt</a>                                 |
| <a href="#">sup</a>                        | Exposant                         | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                           | <a href="#">formulation</a> | <a href="#">formulation</a>                                                                                                                                                                           | <a href="#">globales</a>  | <a href="#">HTML</a><br><a href="#">LEl</a><br><a href="#">ement</a><br><a href="#">nt</a>                                 |
| <a href="#">SVG</a><br><a href="#">svg</a> | Racine SVG                       | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">intégré</a> ; <a href="#">palpable</a> | <a href="#">formulation</a> | par <a href="#">[SVG]</a>                                                                                                                                                                             | par <a href="#">[SVG]</a> | <a href="#">SVG</a><br><a href="#">SVG</a><br><a href="#">Element</a><br><a href="#">t</a>                                 |
| <a href="#">table</a>                      | Tableau                          | <a href="#">flux</a> ; <a href="#">palpable</a>                                                    | <a href="#">couleur</a>     | <a href="#">caption*</a> ; <a href="#">colgroup*</a> ; <a href="#">thead*</a> ; <a href="#">tbody*</a> ; <a href="#">tfoot*</a> ; <a href="#">tr*</a> ; <a href="#">éléments de support de script</a> | <a href="#">globales</a>  | <a href="#">HTML</a><br><a href="#">LTable</a><br><a href="#">Element</a><br><a href="#">t</a>                             |
| <a href="#">tbody</a>                      | Groupe de lignes dans un tableau | aucun                                                                                              | <a href="#">table</a>       | <a href="#">tr</a> ; <a href="#">éléments de support de script</a>                                                                                                                                    | <a href="#">globales</a>  | <a href="#">HTML</a><br><a href="#">LTable</a><br><a href="#">Section</a><br><a href="#">Element</a><br><a href="#">nt</a> |

# Liste des éléments

| Élément                  | Description                 | Catégories                                                                                                                                                                                                                                                          | Parents†                                                                                                                           | Enfants                                                            | Les attributs                                                                                                                                                                                                                                                                                                                                             | Interface                               |
|--------------------------|-----------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------|
| <a href="#">td</a>       | Cellule du tableau          | aucun                                                                                                                                                                                                                                                               | <a href="#">tr</a>                                                                                                                 | <a href="#">couler</a>                                             | <a href="#">globales</a> ; <a href="#">colspan</a> ; <a href="#">rowspan</a> ; <a href="#">headers</a>                                                                                                                                                                                                                                                    | <a href="#">HTMLTableCellElement</a>    |
| <a href="#">template</a> | Modèle                      | <a href="#">métadonnées</a> ; <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">prenant en charge les scripts</a>                                                                                                                                         | <a href="#">métadonnées</a> ; <a href="#">phrases</a> ; <a href="#">prenant en charge les scripts</a> ; <a href="#">colgroupe*</a> | vide                                                               | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                  | <a href="#">HTMLTemplateElement</a>     |
| <a href="#">textarea</a> | Contrôles de texte multiple | <a href="#">flux</a> ; <a href="#">phrase</a> ; <a href="#">interactif</a> ; <a href="#">listé</a> ; <a href="#">étiquetable</a> ; <a href="#">soumissable</a> ; <a href="#">réinitialisable</a> ; <a href="#">associé au formulaire</a> ; <a href="#">palpable</a> | <a href="#">formulation</a>                                                                                                        | <a href="#">texte</a>                                              | <a href="#">globales</a> ; <a href="#">autocomplete cols</a> ; <a href="#">dirname</a> ; <a href="#">disabled</a> ; <a href="#">form</a> ; <a href="#">maxlength</a> ; <a href="#">minlength</a> ; <a href="#">name</a> ; <a href="#">placeholder</a> ; <a href="#">readonly</a> ; <a href="#">required</a> ; <a href="#">rows</a> ; <a href="#">wrap</a> | <a href="#">HTMLTextAreaElement</a>     |
| <a href="#">tfoot</a>    | Groupe de lignes de pied de | aucun                                                                                                                                                                                                                                                               | <a href="#">table</a>                                                                                                              | <a href="#">tr</a> ; <a href="#">éléments de support de script</a> | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                                  | <a href="#">HTMLTableSectionElement</a> |

# Liste des éléments

| Élément      | Description                                | Catégories                                    | Parents†           | Enfants                                          | Les attributs                                                                                   | Interface               |
|--------------|--------------------------------------------|-----------------------------------------------|--------------------|--------------------------------------------------|-------------------------------------------------------------------------------------------------|-------------------------|
|              | page dans un tableau                       |                                               |                    |                                                  |                                                                                                 |                         |
| <u>th</u>    | Cellule d'en-tête de tableau               | <u>interactif</u> *                           | <u>tr</u>          | <u>flux</u> *                                    | <u>globales</u> ; <u>colspan</u> ; <u>rowspan</u> ; <u>headers</u> ; <u>scope</u> ; <u>abbr</u> | HTMLTableCellElement    |
| <u>thead</u> | Groupe de lignes d'en-tête dans un tableau | aucun                                         | <u>table</u>       | <u>tr</u> ; <u>éléments de support de script</u> | <u>globales</u>                                                                                 | HTMLTableSectionElement |
| <u>time</u>  | Équivalent lisible par machine des données | <u>flux</u> ; <u>phrase</u> ; <u>palpable</u> | <u>formulation</u> | <u>formulation</u>                               | <u>globales</u> ; <u>datetime</u>                                                               | HTMLTimeElement         |



# Liste des éléments

| Élément                      | Description                 | Catégories                                                                                    | Parents†                                                                                                                  | Enfants                                                                                                       | Les attributs                                                                                                                                                                               | Interface                                  |
|------------------------------|-----------------------------|-----------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------|
|                              | liée à la date ou à l'heure |                                                                                               |                                                                                                                           |                                                                                                               |                                                                                                                                                                                             |                                            |
| <a href="#"><u>title</u></a> | Titre du document           | <a href="#"><u>métadonnées</u></a>                                                            | <a href="#"><u>head</u></a>                                                                                               | <a href="#"><u>texte</u></a> *                                                                                | <a href="#"><u>globales</u></a>                                                                                                                                                             | <a href="#"><u>HTMLTitleElement</u></a>    |
| <a href="#"><u>tr</u></a>    | Ligne de tableau            | aucun                                                                                         | <a href="#"><u>table</u></a> ; <a href="#"><u>thead</u></a> ; <a href="#"><u>tbody</u></a> ; <a href="#"><u>tfoot</u></a> | <a href="#"><u>th</u></a> *; <a href="#"><u>td</u></a> ; <a href="#"><u>éléments de support de script</u></a> | <a href="#"><u>globales</u></a>                                                                                                                                                             | <a href="#"><u>HTMLTableRowElement</u></a> |
| <a href="#"><u>track</u></a> | Piste de texte chronométrée | aucun                                                                                         | <a href="#"><u>audio</u></a> ; <a href="#"><u>video</u></a>                                                               | vide                                                                                                          | <a href="#"><u>globales</u></a> ; <a href="#"><u>default</u></a> ; <a href="#"><u>kind</u></a> ; <a href="#"><u>label</u></a> ; <a href="#"><u>src</u></a> ; <a href="#"><u>srclang</u></a> | <a href="#"><u>HTMLTrackElement</u></a>    |
| <a href="#"><u>u</u></a>     | Annotation non articulée    | <a href="#"><u>flux</u></a> ; <a href="#"><u>phrasé</u></a> ; <a href="#"><u>palpable</u></a> | <a href="#"><u>formulation</u></a>                                                                                        | <a href="#"><u>formulation</u></a>                                                                            | <a href="#"><u>globales</u></a>                                                                                                                                                             | <a href="#"><u>HTMLElement</u></a>         |
| <a href="#"><u>ul</u></a>    | Liste                       | <a href="#"><u>flux</u></a> ; <a href="#"><u>palpable</u></a> *                               | <a href="#"><u>couleur</u></a>                                                                                            | <a href="#"><u>li</u></a> ; <a href="#"><u>éléments de support de script</u></a>                              | <a href="#"><u>globales</u></a>                                                                                                                                                             | <a href="#"><u>HTMLListElement</u></a>     |

# Liste des éléments

| Élément                                | Description                     | Catégories                                                                                                                      | Parents†                                           | Enfants                                                                             | Les attributs                                                                                                                                                                                                                                                                                                                            | Interface                                                                  |
|----------------------------------------|---------------------------------|---------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------|-------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------|
|                                        |                                 |                                                                                                                                 |                                                    |                                                                                     |                                                                                                                                                                                                                                                                                                                                          | <a href="#">menu</a>                                                       |
| <a href="#">variable</a>               | Variable                        | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                                                        | <a href="#">formulation</a>                        | <a href="#">formulation</a>                                                         | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                 | <a href="#">HTML Element</a>                                               |
| <a href="#">vidéo</a>                  | Lecteur vidéo                   | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">intégré</a> ; <a href="#">interactif</a> ; <a href="#">palpable</a> | <a href="#">formulation</a>                        | <a href="#">source</a> * ; <a href="#">track</a> * ; <a href="#">transparente</a> * | <a href="#">globales</a> ; <a href="#">src</a> ; <a href="#">crossorigin</a> ; <a href="#">poster</a> ; <a href="#">preload</a> ; <a href="#">ad</a> ; <a href="#">autoplay</a> ; <a href="#">playsinline</a> ; <a href="#">loop</a> ; <a href="#">muted</a> ; <a href="#">controls</a> ; <a href="#">width</a> ; <a href="#">height</a> | <a href="#">HTML Video Element</a>                                         |
| <a href="#">web</a>                    | Opportunité de rupture de ligne | <a href="#">flux</a> ; <a href="#">formulation</a>                                                                              | <a href="#">formulation</a>                        | vide                                                                                | <a href="#">globales</a>                                                                                                                                                                                                                                                                                                                 | <a href="#">HTML Element</a>                                               |
| <a href="#">éléments personnalisés</a> | Éléments définis par l'auteur   | <a href="#">flux</a> ; <a href="#">phrasé</a> ; <a href="#">palpable</a>                                                        | <a href="#">flux</a> ; <a href="#">formulation</a> | <a href="#">transparent</a>                                                         | <a href="#">globales</a> ; n'importe lequel, tel que décidé par l'auteur de l'élément                                                                                                                                                                                                                                                    | Fourni par l'auteur de l'élément (hérite de <a href="#">HTML Element</a> ) |

Un astérisque (\*) dans une cellule indique que les règles réelles sont plus compliquées que celles indiquées dans le tableau ci-dessus.

† Les catégories dans la colonne "Parents" font référence aux parents qui répertorient les catégories données dans leur modèle de contenu, et non aux éléments qui se trouvent eux-

mêmes dans ces catégories. Par exemple, la acolonne "Parents" de l'élément indique "phrasing", donc tout élément dont le modèle de contenu contient la catégorie "phrasing" peut être un parent d'un aélément. Étant donné que la catégorie "flow" inclut tous les éléments "phrasing", cela signifie que l' thélément peut être parent d'un aélément.

## Catégories de contenu d'élément

*Cette section est non normative.*

Liste des catégories de contenu d'élément

| Catégorie                      | Éléments                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       | Éléments avec exceptions                                                                                                                                                                   |
|--------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <u>Contenu des métadonnées</u> | <u>base</u> ; <u>link</u> ; <u>meta</u> ; <u>noscript</u> ; <u>script</u> ; <u>style</u> ; <u>template</u> ; <u>title</u>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      | —                                                                                                                                                                                          |
| <u>Contenu du flux</u>         | <u>a</u> ; <u>abbr</u> ; <u>address</u> ; <u>article</u> ; <u>aside</u> ; <u>audio</u> ; <u>b</u> ; <u>bdi</u> ; <u>bdo</u> ; <u>blockquote</u> ; <u>br</u> ; <u>button</u> ; <u>canvas</u> ; <u>cite</u> ; <u>code</u> ; <u>data</u> ; <u>datalist</u> ; <u>del</u> ; <u>details</u> ; <u>dfn</u> ; <u>dialog</u> ; <u>div</u> ; <u>dl</u> ; <u>em</u> ; <u>embed</u> ; <u>fieldset</u> ; <u>figure</u> ; <u>footer</u> ; <u>form</u> ; <u>h1</u> ; <u>h2</u> ; <u>h3</u> ; <u>h4</u> ; <u>h5</u> ; <u>h6</u> ; <u>header</u> ; <u>hgroup</u> ; <u>hr</u> ; <u>i</u> ; <u>iframe</u> ; <u>img</u> ; <u>input</u> ; <u>ins</u> ; <u>kbd</u> ; <u>label</u> ; <u>map</u> ; <u>mark</u> ; <u>MathML</u> <u>math</u> ; <u>menu</u> ; <u>meter</u> ; <u>nav</u> ; <u>noscript</u> ; <u>object</u> ; <u>ol</u> ; <u>output</u> ; <u>p</u> ; <u>picture</u> ; <u>pre</u> ; <u>progress</u> ; <u>q</u> ; <u>ruby</u> ; <u>s</u> ; <u>samp</u> ; <u>script</u> ; <u>section</u> ; <u>select</u> ; <u>slot</u> ; <u>small</u> ; <u>span</u> ; <u>strong</u> ; <u>sub</u> ; <u>sup</u> ; <u>SVG</u> <u>svg</u> ; <u>table</u> ; <u>template</u> ; <u>textarea</u> ; <u>time</u> ; <u>u</u> ; <u>ul</u> ; <u>var</u> ; <u>video</u> ; <u>wbr</u> ; <u>éléments personnalisés autonomes</u> ; <u>Texte</u> | <u>area</u> (si c'est un descendant d'un <u>map</u> élément)<br>; <u>link</u> (si c'est <u>autorisé</u> dans le <u>corps</u> )<br>; <u>main</u> (si c'est un <u>élément hiérarchique</u> ) |

# Liste des catégories de contenu d'élément

| Catégorie                                 | Éléments                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                          | Éléments avec exceptions                                                                                                                                                                         |
|-------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                           |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | <a href="#">ent</a><br><a href="#">correct</a><br><a href="#">main</a> );<br><a href="#">meta</a> (s<br>i<br>l' <a href="#">item</a><br><a href="#">prop</a> att<br>ribut<br>est<br>présen<br>t) |
| <a href="#">Sectionnement du contenu</a>  | <a href="#">article</a> ; <a href="#">aside</a> ; <a href="#">nav</a> ; <a href="#">section</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                   | —                                                                                                                                                                                                |
| <a href="#">Contenu de l'entête</a>       | <a href="#">h1</a> ; <a href="#">h2</a> ; <a href="#">h3</a> ; <a href="#">h4</a> ; <a href="#">h5</a> ; <a href="#">h6</a> ; <a href="#">hgroup</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              | —                                                                                                                                                                                                |
| <a href="#">Contenu de la formulation</a> | <a href="#">a</a> ; <a href="#">abbr</a> ; <a href="#">audio</a> ; <a href="#">b</a> ; <a href="#">bdi</a> ; <a href="#">bdo</a> ; <a href="#">br</a> ; <a href="#">button</a> ; <a href="#">canvas</a> ; <a href="#">cite</a> ; <a href="#">code</a> ; <a href="#">data</a> ; <a href="#">datalist</a> ; <a href="#">del</a> ; <a href="#">dfn</a> ; <a href="#">em</a> ; <a href="#">embed</a> ; <a href="#">i</a> ; <a href="#">iframe</a> ; <a href="#">img</a> ; <a href="#">input</a> ; <a href="#">ins</a> ; <a href="#">kbd</a> ; <a href="#">label</a> ; <a href="#">map</a> ; <a href="#">mark</a> ; <a href="#">MathML</a> <a href="#">math</a> ; <a href="#">meter</a> ; <a href="#">noscript</a> ; <a href="#">object</a> ; <a href="#">output</a> ; <a href="#">picture</a> ; <a href="#">progress</a> ; <a href="#">q</a> ; <a href="#">ruby</a> ; <a href="#">s</a> ; <a href="#">samp</a> ; <a href="#">script</a> ; <a href="#">select</a> ; <a href="#">slot</a> ; <a href="#">small</a> ; <a href="#">span</a> ; <a href="#">strong</a> ; <a href="#">sub</a> ; <a href="#">sup</a> ; <a href="#">SVG</a> <a href="#">svg</a> ; <a href="#">template</a> ; <a href="#">textarea</a> ; <a href="#">time</a> ; <a href="#">u</a> ; <a href="#">var</a> ; <a href="#">video</a> ; <a href="#">wbr</a> ; <a href="#">éléments personnalisés autonomes</a> ; <a href="#">Texte</a> | <a href="#">area</a> (si c'est un descendant d'un <a href="#">map</a> élément)<br>; <a href="#">link</a> (si c'est <a href="#">autorisé</a> dans le                                              |

# Liste des catégories de contenu d'élément

| Catégorie                          | Éléments                                                                                                                                                                                                                                                                                    | Éléments avec exceptions                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                    |                                                                                                                                                                                                                                                                                             | <a href="#">corps</a> )<br>; <a href="#">meta</a> (<br>si<br>l' <a href="#">item</a><br><a href="#">prop</a> <a href="#">att</a><br>ribut<br>est<br>présen<br>t)                                                                                                                                                                                                                                                                                                                                                                                                        |
| <a href="#">Contenu intégré</a>    | <a href="#">audio</a> ; <a href="#">canvas</a> ; <a href="#">embed</a> ; <a href="#">iframe</a> ; <a href="#">img</a> ; <a href="#">MathML</a> <a href="#">math</a> ; <a href="#">object</a> ; <a href="#">picture</a> ; <a href="#">SVG</a><br><a href="#">svg</a> ; <a href="#">video</a> | —                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |
| <a href="#">Contenu interactif</a> | <a href="#">button</a> ; <a href="#">details</a> ; <a href="#">embed</a> ; <a href="#">iframe</a> ; <a href="#">label</a> ; <a href="#">select</a> ; <a href="#">textarea</a>                                                                                                               | <a href="#">a</a> (si<br>l' <a href="#">href</a><br>attribut<br>est<br>présen<br>t); <a href="#">aud</a><br><a href="#">io</a> (si<br>l' <a href="#">cont</a><br><a href="#">rols</a> <a href="#">att</a><br>ribut<br>est<br>présen<br>t); <a href="#">img</a> (<br>si<br>l' <a href="#">usem</a><br><a href="#">ap</a> <a href="#">attri</a><br>but est<br>présen<br>t); <a href="#">inp</a><br><a href="#">ut</a> (si<br>l' <a href="#">type</a><br>attribut<br>n'est p<br>as à l'<br>état <a href="#">M</a><br><a href="#">asqué</a><br>)<br>; <a href="#">video</a> |

# Liste des catégories de contenu d'élément

| Catégorie                       | Éléments                                                                                                                                                                                                                                                           | Éléments avec exceptions                          |
|---------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------|
|                                 |                                                                                                                                                                                                                                                                    | (si l'attribut <code>controls</code> est présent) |
| Éléments associés au formulaire | <code>button</code> ; <code>fieldset</code> ; <code>input</code> ; <code>label</code> ; <code>object</code> ; <code>output</code> ; <code>select</code> ; <code>textarea</code> ; <code>img</code> ; <a href="#">éléments personnalisés associés au formulaire</a> | —                                                 |
| Éléments listés                 | <code>button</code> ; <code>fieldset</code> ; <code>input</code> ; <code>object</code> ; <code>output</code> ; <code>select</code> ; <code>textarea</code> ; <a href="#">éléments personnalisés associés au formulaire</a>                                         | —                                                 |
| Éléments à soumettre            | <code>button</code> ; <code>input</code> ; <code>select</code> ; <code>textarea</code> ; <a href="#">éléments personnalisés associés au formulaire</a>                                                                                                             | —                                                 |
| Éléments réinitialisables       | <code>input</code> ; <code>output</code> ; <code>select</code> ; <code>textarea</code> ; <a href="#">éléments personnalisés associés au formulaire</a>                                                                                                             | —                                                 |
| Éléments                        | <code>button</code> ; <code>fieldset</code> ; <code>input</code> ; <code>output</code> ; <code>select</code> ; <code>textarea</code>                                                                                                                               | —                                                 |

# Liste des catégories de contenu d'élément

| Catégorie                                                  | Éléments                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                     | Éléments avec exceptions                                                                                                                                                                                                                      |
|------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <a href="#">hérédités de la capitalisation automatique</a> |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                              |                                                                                                                                                                                                                                               |
| <a href="#">Éléments étiquetables</a>                      | <a href="#">button</a> ; <a href="#">input</a> ; <a href="#">meter</a> ; <a href="#">output</a> ; <a href="#">progress</a> ; <a href="#">select</a> ; <a href="#">textarea</a> ; <a href="#">éléments personnalisés associés au formulaire</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               | —                                                                                                                                                                                                                                             |
| <a href="#">Contenu palpable</a>                           | <a href="#">a</a> ; <a href="#">abbr</a> ; <a href="#">address</a> ; <a href="#">article</a> ; <a href="#">aside</a> ; <a href="#">b</a> ; <a href="#">bdi</a> ; <a href="#">bdo</a> ; <a href="#">blockquote</a> ; <a href="#">button</a> ; <a href="#">canva</a> ; <a href="#">s</a> ; <a href="#">cite</a> ; <a href="#">code</a> ; <a href="#">data</a> ; <a href="#">del</a> ; <a href="#">details</a> ; <a href="#">dfn</a> ; <a href="#">div</a> ; <a href="#">em</a> ; <a href="#">embed</a> ; <a href="#">fieldset</a> ; <a href="#">figure</a> ; <a href="#">footer</a> ; <a href="#">form</a> ; <a href="#">h1</a> ; <a href="#">h2</a> ; <a href="#">h3</a> ; <a href="#">h4</a> ; <a href="#">h5</a> ; <a href="#">h6</a> ; <a href="#">header</a> ; <a href="#">hgroup</a> ; <a href="#">i</a> ; <a href="#">iframe</a> ; <a href="#">img</a> ; <a href="#">ins</a> ; <a href="#">k</a> ; <a href="#">bd</a> ; <a href="#">label</a> ; <a href="#">main</a> ; <a href="#">map</a> ; <a href="#">mark</a> ; <a href="#">MathML</a> <a href="#">math</a> ; <a href="#">meter</a> ; <a href="#">nav</a> ; <a href="#">object</a> ; <a href="#">output</a> ; <a href="#">p</a> ; <a href="#">picture</a> ; <a href="#">pre</a> ; <a href="#">progress</a> ; <a href="#">q</a> ; <a href="#">ruby</a> ; <a href="#">s</a> ; <a href="#">samp</a> ; <a href="#">section</a> ; <a href="#">select</a> ; <a href="#">small</a> ; <a href="#">span</a> ; <a href="#">s</a> ; <a href="#">trong</a> ; <a href="#">sub</a> ; <a href="#">sup</a> ; <a href="#">SVG</a> <a href="#">svg</a> ; <a href="#">table</a> ; <a href="#">textarea</a> ; <a href="#">time</a> ; <a href="#">u</a> ; <a href="#">var</a> ; <a href="#">video</a> ; <a href="#">éléments personnalisés autonomes</a> | <a href="#">audio</a> (si l' <a href="#">cont</a> <a href="#">rol</a> s attribut est présent); <a href="#">dl</a> (si les enfants de l'élément incluent au moins un groupe nominal-valeur); <a href="#">input</a> (si l' <a href="#">type</a> |

Liste des catégories de contenu d'élément

| Catégorie | Éléments | Éléments avec exceptions                                                                                                                                                                                                                                                                                                                                                                                                       |
|-----------|----------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|           |          | <p>attribut n'est pas à l'état <a href="#">Masqué</a> )</p> <p>; <a href="#">menu</a>( si les enfants de l'élément incluent au moins un <a href="#">lié</a> élément )</p> <p>; <a href="#">ol</a>(si les enfants de l'élément incluent au moins un <a href="#">lié</a> élément )</p> <p>; <a href="#">ul</a>(si les enfants de l'élément incluent au moins un <a href="#">lié</a> élément )</p> <p>; <a href="#">Texte</a></p> |



## Liste des catégories de contenu d'élément

| Catégorie                                     | Éléments                                    | Éléments avec exceptions                                   |
|-----------------------------------------------|---------------------------------------------|------------------------------------------------------------|
|                                               |                                             | qui n'est pas <a href="#">un espace entre les éléments</a> |
| <a href="#">Éléments de support de script</a> | <code>script</code> ; <code>template</code> | —                                                          |

## Les attributs

*Cette section est non normative.*

### Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut          | Éléments        | Description                                                                                                            | Valeur                  |
|-------------------|-----------------|------------------------------------------------------------------------------------------------------------------------|-------------------------|
| <code>abbr</code> | <code>th</code> | Étiquette alternative à utiliser pour la cellule d'en-tête lors du référencement de la cellule dans d'autres contextes | <a href="#">Texte</a> * |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut        | Éléments                                                           | Description                                                                                                | Valeur                                                                                                                                                                            |
|-----------------|--------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| accept          | <a href="#">input</a>                                              | Conseil pour le type de fichier attendu dans <a href="#">les contrôles de téléchargement de fichiers</a>   | <a href="#">Ensemble de jetons séparés par des virgules</a> * composé de <a href="#">chaînes de type MIME valides sans paramètres</a> ou audio/*, video/*, ou image/*             |
| accept-charset  | <a href="#">form</a>                                               | Encodages de caractères à utiliser pour <a href="#">la soumission de formulaire</a>                        | <a href="#">Correspondance ASCII insensible à la casse</a> pour " UTF-8"                                                                                                          |
| accesskey       | <a href="#">Éléments HTML</a>                                      | Raccourci clavier pour activer ou focaliser l'élément                                                      | <a href="#">Ensemble ordonné de jetons uniques séparés par des espaces</a> , dont aucun n'est <a href="#">identique</a> à un autre, chacun composé d'un point de code de longueur |
| action          | <a href="#">form</a>                                               | <a href="#">URL</a> à utiliser pour <a href="#">la soumission du formulaire</a>                            | <a href="#">URL valide non vide potentiellement entourée d'espaces</a>                                                                                                            |
| allow           | <a href="#">iframe</a>                                             | <a href="#">Politique d'autorisations</a> à appliquer au <a href="#">iframe</a> contenu de                 | <a href="#">Politique d'autorisations sérialisées</a>                                                                                                                             |
| allowfullscreen | <a href="#">iframe</a>                                             | S'il faut autoriser <a href="#">iframe</a> l'utilisation du contenu de <a href="#">requestFullscreen()</a> | <a href="#">Attribut booléen</a>                                                                                                                                                  |
| alt             | <a href="#">area</a> ; <a href="#">img</a> ; <a href="#">input</a> | Texte de remplacement à utiliser lorsque les images ne                                                     | <a href="#">Texte</a> *                                                                                                                                                           |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut       | Éléments                                                                  | Description                                                                                                                                   | Valeur                                                                                                                                                         |
|----------------|---------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                |                                                                           | sont pas disponibles                                                                                                                          |                                                                                                                                                                |
| as             | <a href="#">link</a>                                                      | <a href="#">Destination potentielle</a> d'une demande de préchargement (pour <code>rel="preload"</code> et <code>rel="modulepreload"</code> ) | <a href="#">Destination potentielle</a> , pour <code>rel="preload"</code> ; <a href="#">destination de type script</a> , pour <code>rel="modulepreload"</code> |
| async          | <a href="#">script</a>                                                    | Exécuter le script lorsqu'il est disponible, sans bloquer lors de la récupération                                                             | <a href="#">Attribut booléen</a>                                                                                                                               |
| autocapitalize | <a href="#">Éléments HTML</a>                                             | Comportement de capitalisation automatique recommandé (pour les méthodes de saisie prises en charge)                                          | <code>"on"; "off"; "none"; "sentences"; "words"; "characters"</code>                                                                                           |
| autocomplete   | <a href="#">form</a>                                                      | Paramètre par défaut de la fonction de remplissage automatique pour les contrôles du formulaire                                               | <code>"on"; "off"</code>                                                                                                                                       |
| autocomplete   | <a href="#">input</a> ; <a href="#">select</a> ; <a href="#">textarea</a> | Astuce pour la fonction de remplissage automatique du formulaire                                                                              | <a href="#">Nom du champ de saisie automatique</a> et jetons associés*                                                                                         |
| autofocus      | <a href="#">Éléments HTML</a>                                             | Focaliser automatique                                                                                                                         | <a href="#">Attribut booléen</a>                                                                                                                               |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut | Éléments                                                                                   | Description                                                                                                       | Valeur                                                                           |
|----------|--------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------|
|          |                                                                                            | ment l'élément lors du chargement de la page                                                                      |                                                                                  |
| autoplay | <a href="#">audio</a> ; <a href="#">video</a>                                              | Indice que la <a href="#">ressource multimédia</a> peut être démarrée automatiquement lorsque la page est chargée | <a href="#">Attribut booléen</a>                                                 |
| blocking | <a href="#">link</a> ; <a href="#">script</a> ; <a href="#">style</a>                      | Si l'élément est <a href="#">potentiellement bloquant le rendu</a>                                                | <a href="#">Ensemble non ordonné de jetons uniques séparés par des espaces</a> * |
| charset  | <a href="#">meta</a>                                                                       | <a href="#">Déclaration d'encodage de caractères</a>                                                              | "utf-8"                                                                          |
| checked  | <a href="#">input</a>                                                                      | Si le contrôle est coché                                                                                          | <a href="#">Attribut booléen</a>                                                 |
| cite     | <a href="#">blockquote</a> ; <a href="#">del</a> ; <a href="#">ins</a> ; <a href="#">q</a> | Lien vers la source de la citation ou plus d'informations sur la modification                                     | <a href="#">URL valide potentiellement entourée d'espaces</a>                    |
| class    | <a href="#">Éléments HTML</a>                                                              | Classes auxquelles appartient l'élément                                                                           | <a href="#">Ensemble de jetons séparés par des espaces</a>                       |
| color    | <a href="#">link</a>                                                                       | Couleur à utiliser lors de la personnalisation de l'icône d'un site                                               | CSS <a href="#">&lt;couleur&gt;</a>                                              |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut                     | Éléments                                                                                             | Description                                                              | Valeur                                                                                                                                                                                               |
|------------------------------|------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                              |                                                                                                      | (pour <code>rel="mask-icon"</code> )                                     |                                                                                                                                                                                                      |
| <code>cols</code>            | <code>textarea</code>                                                                                | Nombre maximum de caractères par ligne                                   | <a href="#">Entier non négatif valide</a> supérieur à zéro                                                                                                                                           |
| <code>colspan</code>         | <code>td</code> ; <code>th</code>                                                                    | Nombre de colonnes sur lesquelles la cellule doit s'étendre              | <a href="#">Entier non négatif valide</a> supérieur à zéro                                                                                                                                           |
| <code>content</code>         | <code>meta</code>                                                                                    | Valeur de l'élément                                                      | <a href="#">Texte</a> *                                                                                                                                                                              |
| <code>contenteditable</code> | <a href="#">Éléments HTML</a>                                                                        | Si l'élément est modifiable                                              | " true"; " false"                                                                                                                                                                                    |
| <code>controls</code>        | <code>audio</code> ; <code>video</code>                                                              | Afficher les contrôles de l'agent utilisateur                            | <a href="#">Attribut booléen</a>                                                                                                                                                                     |
| <code>coords</code>          | <code>area</code>                                                                                    | Coordonnées de la forme à créer dans une <a href="#">image cliquable</a> | <a href="#">Liste valide de nombres à virgule flottante</a> *                                                                                                                                        |
| <code>crossorigin</code>     | <code>audio</code> ; <code>img</code> ; <code>link</code> ; <code>script</code> ; <code>video</code> | Comment l'élément gère les requêtes crossorigin                          | " <code>anonymous</code> "; " <code>use-credentials</code> "                                                                                                                                         |
| <code>data</code>            | <code>object</code>                                                                                  | Adresse de la ressource                                                  | <a href="#">URL valide non vide potentiellement entourée d'espaces</a>                                                                                                                               |
| <code>datetime</code>        | <code>del</code> ; <code>ins</code>                                                                  | Date et (éventuellement) heure du changement                             | <a href="#">Chaîne de date valide avec heure facultative</a>                                                                                                                                         |
| <code>datetime</code>        | <code>time</code>                                                                                    | Valeur lisible par machine                                               | <a href="#">Chaîne de mois valide</a> , <a href="#">chaîne de date valide</a> , <a href="#">chaîne de date sans année valide</a> , <a href="#">chaîne d'heure valide</a> , <a href="#">chaîne de</a> |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut | Éléments                                         | Description                                                                                                                      | Valeur                                                                                                                                                                                                                                                                                                  |
|----------|--------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|          |                                                  |                                                                                                                                  | <a href="#">date et d'heure locale valide</a> , <a href="#">chaîne de décalage de fuseau horaire valide</a> , <a href="#">chaîne de date et d'heure globale valide</a> , <a href="#">chaîne de semaine valide</a> , <a href="#">entier non négatif valide</a> ou <a href="#">valide chaîne de durée</a> |
| decoding | <a href="#">img</a>                              | Indice de décodage à utiliser lors du traitement de cette image pour la présentation                                             | " <a href="#">sync</a> " ; " <a href="#">async</a> " ; " <a href="#">auto</a> "                                                                                                                                                                                                                         |
| default  | <a href="#">track</a>                            | Activer la piste si aucune autre <a href="#">piste de texte</a> n'est plus adaptée                                               | <a href="#">Attribut booléen</a>                                                                                                                                                                                                                                                                        |
| defer    | <a href="#">script</a>                           | Différer l'exécution du script                                                                                                   | <a href="#">Attribut booléen</a>                                                                                                                                                                                                                                                                        |
| dir      | <a href="#">Éléments HTML</a>                    | <a href="#">La directionnalité du texte</a> de l'élément                                                                         | " <a href="#">ltr</a> " ; " <a href="#">rtl</a> " ; " <a href="#">auto</a> "                                                                                                                                                                                                                            |
| dir      | <a href="#">bdo</a>                              | <a href="#">La directionnalité du texte</a> de l'élément                                                                         | " <a href="#">ltr</a> " ; " <a href="#">rtl</a> "                                                                                                                                                                                                                                                       |
| dirname  | <a href="#">input</a> ; <a href="#">textarea</a> | Nom du contrôle de formulaire à utiliser pour envoyer la <a href="#">directionnalité</a> de l'élément lors de <a href="#">la</a> | <a href="#">Texte</a> *                                                                                                                                                                                                                                                                                 |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut     | Éléments)                                                                                                                                                                                                              | Description                                                                                                                 | Valeur                                                                                                                                                                                    |
|--------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|              |                                                                                                                                                                                                                        | <a href="#">soumission du formulaire</a>                                                                                    |                                                                                                                                                                                           |
| disabled     | <a href="#">button</a> ; <a href="#">input</a> ; <a href="#">optgroup</a> ; <a href="#">option</a> ; <a href="#">select</a> ; <a href="#">textarea</a> ; <a href="#">éléments personnalisés associés au formulaire</a> | Si le contrôle de formulaire est désactivé                                                                                  | <a href="#">Attribut booléen</a>                                                                                                                                                          |
| disabled     | <a href="#">fieldset</a>                                                                                                                                                                                               | Si les contrôles de formulaire descendants, à l'exception de ceux à l'intérieur de <a href="#">legend</a> , sont désactivés | <a href="#">Attribut booléen</a>                                                                                                                                                          |
| disabled     | <a href="#">link</a>                                                                                                                                                                                                   | Si le lien est désactivé                                                                                                    | <a href="#">Attribut booléen</a>                                                                                                                                                          |
| download     | <a href="#">a</a> ; <a href="#">area</a>                                                                                                                                                                               | S'il faut télécharger la ressource au lieu d'y accéder, et son nom de fichier si c'est le cas                               | Texte                                                                                                                                                                                     |
| draggable    | <a href="#">Éléments HTML</a>                                                                                                                                                                                          | Si l'élément est déplaçable                                                                                                 | "true"; "false"                                                                                                                                                                           |
| enctype      | <a href="#">form</a>                                                                                                                                                                                                   | <a href="#">Type d'encodage de la liste d'entrées</a> à utiliser pour <a href="#">la soumission du formulaire</a>           | " <a href="#">application/x-www-form-urlencoded</a> "; " <a href="#">multipart/form-data</a> "; " <a href="#">text/plain</a> "                                                            |
| enterkeyhint | <a href="#">Éléments HTML</a>                                                                                                                                                                                          | Conseil pour sélectionner une action de touche Entrée                                                                       | " <a href="#">enter</a> "; " <a href="#">done</a> "; " <a href="#">go</a> "; " <a href="#">next</a> "; " <a href="#">previous</a> "; " <a href="#">search</a> "; " <a href="#">send</a> " |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut       | Éléments                                                                                                                                                                                                                                        | Description                                                                                                       | Valeur                                                                                                                         |
|----------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------|
| fetchpriority  | <a href="#">img</a> ; <a href="#">link</a> ; <a href="#">script</a>                                                                                                                                                                             | Définit la <a href="#">priorité</a> de s <a href="#">récupérations</a> initiées par l'élément                     | " <a href="#">auto</a> "; " <a href="#">high</a> "; " <a href="#">low</a> "                                                    |
| for            | <a href="#">label</a>                                                                                                                                                                                                                           | Associer l'étiquette au contrôle de formulaire                                                                    | <a href="#">pièce d'identité</a> *                                                                                             |
| for            | <a href="#">output</a>                                                                                                                                                                                                                          | Spécifie les contrôles à partir desquels la sortie a été calculée                                                 | <a href="#">Ensemble non ordonné de jetons uniques séparés par des espaces</a> et composés d'identifiants*                     |
| form           | <a href="#">button</a> ; <a href="#">fieldset</a> ; <a href="#">input</a> ; <a href="#">object</a> ; <a href="#">output</a> ; <a href="#">select</a> ; <a href="#">textarea</a> ; <a href="#">éléments personnalisés associés au formulaire</a> | Associe l'élément à un <a href="#">form</a> élément                                                               | <a href="#">pièce d'identité</a> *                                                                                             |
| formaction     | <a href="#">button</a> ; <a href="#">input</a>                                                                                                                                                                                                  | <a href="#">URL</a> à utiliser pour <a href="#">la soumission du formulaire</a>                                   | <a href="#">URL valide non vide potentiellement entourée d'espaces</a>                                                         |
| formencoding   | <a href="#">button</a> ; <a href="#">input</a>                                                                                                                                                                                                  | <a href="#">Type d'encodage de la liste d'entrées</a> à utiliser pour <a href="#">la soumission du formulaire</a> | " <a href="#">application/x-www-form-urlencoded</a> "; " <a href="#">multipart/form-data</a> "; " <a href="#">text/plain</a> " |
| formmethod     | <a href="#">button</a> ; <a href="#">input</a>                                                                                                                                                                                                  | Variante à utiliser pour <a href="#">la soumission de formulaire</a>                                              | "GET"; "POST"; "dialog"                                                                                                        |
| formnovalidate | <a href="#">button</a> ; <a href="#">input</a>                                                                                                                                                                                                  | Contourner la validation du contrôle du formulaire pour <a href="#">la</a>                                        | <a href="#">Attribut booléen</a>                                                                                               |



Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut   | Éléments                                                                                                                                                                                                                     | Description                                                | Valeur                                                                                                                                                                                 |
|------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|            |                                                                                                                                                                                                                              | <a href="#">soumission du formulaire</a>                   |                                                                                                                                                                                        |
| formtarget | <a href="#">button</a> ; <a href="#">input</a>                                                                                                                                                                               | Navigable pour <a href="#">la soumission du formulaire</a> | <a href="#">Nom de cible navigable ou mot-clé valide</a>                                                                                                                               |
| headers    | <a href="#">td</a> ; <a href="#">th</a>                                                                                                                                                                                      | Les cellules d'en-tête de cette cellule                    | <a href="#">Ensemble non ordonné de jetons uniques séparés par des espaces</a> et composés d'identifiants*                                                                             |
| height     | <a href="#">canvas</a> ; <a href="#">embed</a> ; <a href="#">iframe</a> ; <a href="#">img</a> ; <a href="#">input</a> ; <a href="#">object</a> ; <a href="#">source</a> (en <a href="#">picture</a> ); <a href="#">video</a> | Dimensions verticales                                      | <a href="#">Entier non négatif valide</a>                                                                                                                                              |
| hidden     | <a href="#">Éléments HTML</a>                                                                                                                                                                                                | Si l'élément est pertinent                                 | " <a href="#">until-found</a> " ; " <a href="#">hidden</a> " ; la chaîne vide                                                                                                          |
| high       | <a href="#">meter</a>                                                                                                                                                                                                        | Limite basse de la plage haute                             | <a href="#">Nombre à virgule flottante valide</a> *                                                                                                                                    |
| href       | <a href="#">a</a> ; <a href="#">area</a>                                                                                                                                                                                     | Adresse du <a href="#">lien hypertexte</a>                 | <a href="#">URL valide potentiellement entourée d'espaces</a>                                                                                                                          |
| href       | <a href="#">link</a>                                                                                                                                                                                                         | Adresse du <a href="#">lien hypertexte</a>                 | <a href="#">URL valide non vide potentiellement entourée d'espaces</a>                                                                                                                 |
| href       | <a href="#">base</a>                                                                                                                                                                                                         | <a href="#">URL de base du document</a>                    | <a href="#">URL valide potentiellement entourée d'espaces</a>                                                                                                                          |
| hreflang   | <a href="#">a</a> ; <a href="#">link</a>                                                                                                                                                                                     | Langue de la ressource liée                                | Balise de langue BCP 47 valide                                                                                                                                                         |
| http-equiv | <a href="#">meta</a>                                                                                                                                                                                                         | Directive pragmatique                                      | " <a href="#">content-type</a> " ; " <a href="#">default-style</a> " ; " <a href="#">refresh</a> " ; " <a href="#">x-ua-compatible</a> " ; " <a href="#">content-security-policy</a> " |
| id         | <a href="#">Éléments HTML</a>                                                                                                                                                                                                | L' <a href="#">identifiant de l'élément</a>                | <a href="#">Texte</a> *                                                                                                                                                                |
| imagesizes | <a href="#">link</a>                                                                                                                                                                                                         | Tailles d'image pour différentes mises en page             | <a href="#">Liste de taille de source valide</a>                                                                                                                                       |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut    | Éléments                                      | Description                                                                                                                                          | Valeur                                                                                                                                                                                                                |
|-------------|-----------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|             |                                               | (pour <a href="#">rel="preload"</a> )                                                                                                                |                                                                                                                                                                                                                       |
| imagesrcset | <a href="#">link</a>                          | Images à utiliser dans différentes situations, par exemple, écrans haute résolution, petits moniteurs, etc.<br>(pour <a href="#">rel="preload"</a> ) | Liste séparée par des <a href="#">virgules des chaînes candidates d'image</a>                                                                                                                                         |
| inert       | <a href="#">Éléments HTML</a>                 | Si l'élément est <a href="#">inerte</a> .                                                                                                            | <a href="#">Attribut booléen</a>                                                                                                                                                                                      |
| inputmode   | <a href="#">Éléments HTML</a>                 | Conseil pour sélectionner une modalité d'entrée                                                                                                      | " <a href="#">none</a> "; " <a href="#">text</a> "; " <a href="#">tel</a> "; " <a href="#">email</a> "; " <a href="#">url</a> "; " <a href="#">numeric</a> "; " <a href="#">decimal</a> "; " <a href="#">search</a> " |
| integrity   | <a href="#">link</a> ; <a href="#">script</a> | Métadonnées d'intégrité utilisées dans les contrôles <i>d'intégrité des sous-ressources</i> [ <a href="#">SRI</a> ]                                  | <a href="#">Texte</a>                                                                                                                                                                                                 |
| is          | <a href="#">Éléments HTML</a>                 | Crée un <a href="#">élément intégré personnalisé</a>                                                                                                 | <a href="#">Nom d'élément personnalisé valide</a> d'un <a href="#">élément intégré personnalisé défini</a>                                                                                                            |
| ismap       | <a href="#">img</a>                           | Si l'image est une image cliquable côté serveur                                                                                                      | <a href="#">Attribut booléen</a>                                                                                                                                                                                      |
| itemid      | <a href="#">Éléments HTML</a>                 | <a href="#">Identificateur global</a> d'un élément de microdonnées                                                                                   | <a href="#">URL valide potentiellement entourée d'espaces</a>                                                                                                                                                         |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut  | Éléments                                                                  | Description                                              | Valeur                                                                                                                                        |
|-----------|---------------------------------------------------------------------------|----------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------|
| itemprop  | <a href="#">Éléments HTML</a>                                             | Noms de propriété d'un élément de microdonnées           | Ensemble non ordonné de jetons uniques séparés par des espaces constitués d' URL absolues valides , de noms de propriété définis ou de texte* |
| itemref   | <a href="#">Éléments HTML</a>                                             | Éléments référencés                                      | Ensemble non ordonné de jetons uniques séparés par des espaces et composés d'identifiants*                                                    |
| itemscope | <a href="#">Éléments HTML</a>                                             | Introduit un élément de microdonnées                     | Attribut booléen                                                                                                                              |
| itemtype  | <a href="#">Éléments HTML</a>                                             | Types d'éléments d'un élément de microdonnées            | Ensemble non ordonné de jetons uniques séparés par des espaces constitués d' URL absolues valides *                                           |
| kind      | <a href="#">track</a>                                                     | Le type de piste de texte                                | " subtitles" ; " captions" ; " descriptions" ; " chapters" ; " metadata"                                                                      |
| label     | <a href="#">optgroup</a> ; <a href="#">option</a> ; <a href="#">track</a> | Étiquette visible par l'utilisateur                      | Texte                                                                                                                                         |
| lang      | <a href="#">Éléments HTML</a>                                             | Langue de l'élément                                      | Balise de langue BCP 47 valide ou chaîne vide                                                                                                 |
| list      | <a href="#">input</a>                                                     | Liste des options de saisie semi-automatique             | pièce d'identité *                                                                                                                            |
| loading   | <a href="#">iframe</a> ; <a href="#">img</a>                              | Utilisé lors de la détermination du report de chargement | " lazy" ; " eager"                                                                                                                            |
| loop      | <a href="#">audio</a> ; <a href="#">video</a>                             | S'il faut boucler                                        | Attribut booléen                                                                                                                              |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut  | Éléments                                                                                                                                                                                                               | Description                                                             | Valeur                                                                          |
|-----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------|---------------------------------------------------------------------------------|
|           |                                                                                                                                                                                                                        | la <a href="#">ressource multimédia</a>                                 |                                                                                 |
| low       | <a href="#">meter</a>                                                                                                                                                                                                  | Limite haute de la plage basse                                          | <a href="#">Nombre à virgule flottante valide</a> *                             |
| max       | <a href="#">input</a>                                                                                                                                                                                                  | Valeur maximum                                                          | Varie*                                                                          |
| max       | <a href="#">meter</a> ; <a href="#">progress</a>                                                                                                                                                                       | Limite supérieure de la plage                                           | <a href="#">Nombre à virgule flottante valide</a> *                             |
| maxlength | <a href="#">input</a> ; <a href="#">textarea</a>                                                                                                                                                                       | <a href="#">Longueur</a> maximale de la valeur                          | <a href="#">Entier non négatif valide</a>                                       |
| media     | <a href="#">link</a> ; <a href="#">meta</a> ; <a href="#">source</a> (en <a href="#">picture</a> ); <a href="#">style</a>                                                                                              | Médias applicables                                                      | <a href="#">Liste de requêtes multimédia valide</a>                             |
| method    | <a href="#">form</a>                                                                                                                                                                                                   | Variante à utiliser pour la <a href="#">soumission de formulaire</a>    | " <a href="#">GET</a> " ; " <a href="#">POST</a> " ; " <a href="#">dialog</a> " |
| min       | <a href="#">input</a>                                                                                                                                                                                                  | Valeur minimum                                                          | Varie*                                                                          |
| min       | <a href="#">meter</a>                                                                                                                                                                                                  | Limite inférieure de la plage                                           | <a href="#">Nombre à virgule flottante valide</a> *                             |
| minlength | <a href="#">input</a> ; <a href="#">textarea</a>                                                                                                                                                                       | <a href="#">Longueur</a> minimale de la valeur                          | <a href="#">Entier non négatif valide</a>                                       |
| multiple  | <a href="#">input</a> ; <a href="#">select</a>                                                                                                                                                                         | S'il faut autoriser plusieurs valeurs                                   | <a href="#">Attribut booléen</a>                                                |
| muted     | <a href="#">audio</a> ; <a href="#">video</a>                                                                                                                                                                          | S'il faut désactiver la <a href="#">ressource multimédia</a> par défaut | <a href="#">Attribut booléen</a>                                                |
| name      | <a href="#">button</a> ; <a href="#">fieldset</a> ; <a href="#">input</a> ; <a href="#">output</a> ; <a href="#">select</a> ; <a href="#">textarea</a> ; <a href="#">éléments personnalisés associés au formulaire</a> | Nom de l'élément à utiliser pour la                                     | <a href="#">Texte</a> *                                                         |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut | Éléments                                  | Description                                                                                                     | Valeur                                                   |
|----------|-------------------------------------------|-----------------------------------------------------------------------------------------------------------------|----------------------------------------------------------|
|          |                                           | <a href="#">soumission du formulaire</a> et dans l' <code>form.elements</code> API                              |                                                          |
| name     | <code>form</code>                         | Nom du formulaire à utiliser dans l' <code>document.forms</code> API                                            | <a href="#">Texte</a> *                                  |
| name     | <code>iframe</code> ; <code>object</code> | Nom du <a href="#">contenu navigable</a>                                                                        | <a href="#">Nom de cible navigable ou mot-clé valide</a> |
| name     | <code>map</code>                          | Nom de l' <a href="#">image cliquable</a> à <a href="#">référer</a> à partir de l' <code>usemap</code> attribut | <a href="#">Texte</a> *                                  |
| name     | <code>meta</code>                         | Nom des métadonnées                                                                                             | <a href="#">Texte</a> *                                  |
| name     | <code>slot</code>                         | Nom de l'emplacement de l'arbre d'ombre                                                                         | <a href="#">Texte</a>                                    |
| nomodule | <code>script</code>                       | Empêche l'exécution dans les agents utilisateurs qui prennent en charge <a href="#">les scripts de module</a>   | <a href="#">Attribut booléen</a>                         |
| nonce    | <a href="#">Éléments HTML</a>             | Nonce cryptographique utilisé dans les vérifications                                                            | <a href="#">Texte</a>                                    |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut    | Éléments                                         | Description                                                                                         | Valeur                                                                                                      |
|-------------|--------------------------------------------------|-----------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------|
|             |                                                  | de la politique de sécurité du contenu <a href="#">[CSP]</a>                                        |                                                                                                             |
| novalidate  | <a href="#">form</a>                             | Contourner la validation du contrôle du formulaire pour <a href="#">la soumission du formulaire</a> | <a href="#">Attribut booléen</a>                                                                            |
| open        | <a href="#">details</a>                          | Si les détails sont visibles                                                                        | <a href="#">Attribut booléen</a>                                                                            |
| open        | <a href="#">dialog</a>                           | Si la boîte de dialogue s'affiche                                                                   | <a href="#">Attribut booléen</a>                                                                            |
| optimum     | <a href="#">meter</a>                            | Valeur optimale en jauge                                                                            | <a href="#">Nombre à virgule flottante valide</a> *                                                         |
| pattern     | <a href="#">input</a>                            | Modèle à faire correspondre à la valeur du contrôle de formulaire                                   | Expression régulière correspondant à la production <a href="#">du modèle JavaScript</a>                     |
| ping        | <a href="#">a</a> ; <a href="#">area</a>         | <a href="#">URL</a> à pinger                                                                        | <a href="#">Ensemble de jetons séparés par des espaces</a> composé d' <a href="#">URL valides non vides</a> |
| placeholder | <a href="#">input</a> ; <a href="#">textarea</a> | Étiquette visible par l'utilisateur à placer dans le contrôle du formulaire                         | <a href="#">Texte</a> *                                                                                     |
| playsinline | <a href="#">video</a>                            | Encouragez l'agent utilisateur à afficher le contenu vidéo dans la                                  | <a href="#">Attribut booléen</a>                                                                            |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut            | Éléments                                         | Description                                                                                                 | Valeur                                                                           |
|---------------------|--------------------------------------------------|-------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------|
|                     |                                                  | zone de lecture de l'élément                                                                                |                                                                                  |
| popover             | <a href="#">Éléments HTML</a>                    | Fait de l'élément un élément <a href="#">popover</a>                                                        | " <a href="#">auto</a> "; " <a href="#">manual</a> ";                            |
| popoverhidetarget   | <a href="#">input</a> ; <a href="#">button</a>   | Masque l'élément <a href="#">popover</a> spécifié lorsqu'il est cliqué                                      | ID de l'élément à masquer                                                        |
| popovershowtarget   | <a href="#">input</a> ; <a href="#">button</a>   | Affiche l'élément <a href="#">popover</a> spécifié lorsqu'il est cliqué                                     | ID de l'élément à afficher                                                       |
| popovertoggletarget | <a href="#">input</a> ; <a href="#">button</a>   | Bascule l'élément <a href="#">popover</a> spécifié lorsqu'il est cliqué                                     | ID de l'élément à basculer                                                       |
| poster              | <a href="#">video</a>                            | Image d'affiche à afficher avant la lecture de la vidéo                                                     | <a href="#">URL valide non vide potentiellement entourée d'espaces</a>           |
| preload             | <a href="#">audio</a> ; <a href="#">video</a>    | Indique la quantité de mémoire tampon dont la <a href="#">ressource multimédia</a> aura probablement besoin | " <a href="#">none</a> "; " <a href="#">metadata</a> "; " <a href="#">auto</a> " |
| readonly            | <a href="#">input</a> ; <a href="#">textarea</a> | Autoriser ou non la valeur à être modifiée par l'utilisateur                                                | <a href="#">Attribut booléen</a>                                                 |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut       | Éléments                                                                                                                                | Description                                                                                                               | Valeur                                                                           |
|----------------|-----------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------|
| readonly       | <a href="#">éléments personnalisés associés au formulaire</a>                                                                           | Affects <a href="#">willvalidate</a> , plus tout comportement ajouté par l'auteur de l'élément personnalisé               | <a href="#">Attribut booléen</a>                                                 |
| referrerpolicy | <a href="#">a</a> ; <a href="#">area</a> ; <a href="#">iframe</a> ; <a href="#">img</a> ; <a href="#">link</a> ; <a href="#">script</a> | <a href="#">Politique de référence</a> pour <a href="#">les récupérations</a> initiées par l'élément                      | <a href="#">Politique de parrainage</a>                                          |
| rel            | <a href="#">a</a> ; <a href="#">area</a>                                                                                                | Relation entre l'emplacement dans le document contenant le <a href="#">lien hypertexte</a> et la ressource de destination | <a href="#">Ensemble non ordonné de jetons uniques séparés par des espaces</a> * |
| rel            | <a href="#">link</a>                                                                                                                    | Relation entre le document contenant le <a href="#">lien hypertexte</a> et la ressource de destination                    | <a href="#">Ensemble non ordonné de jetons uniques séparés par des espaces</a> * |
| required       | <a href="#">input</a> ; <a href="#">select</a> ; <a href="#">textarea</a>                                                               | Si le contrôle est requis pour <a href="#">la soumission du formulaire</a>                                                | <a href="#">Attribut booléen</a>                                                 |
| reversed       | <a href="#">ol</a>                                                                                                                      | Numéroter la liste à l'envers                                                                                             | <a href="#">Attribut booléen</a>                                                 |



Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut | Éléments                                | Description                                                      | Valeur                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                               |
|----------|-----------------------------------------|------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| rows     | <a href="#">textarea</a>                | Nombre de lignes à afficher                                      | <a href="#">Entier non négatif valide</a> supérieur à zéro                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
| rowspan  | <a href="#">td</a> ; <a href="#">th</a> | Nombre de lignes sur lesquelles la cellule doit s'étendre        | <a href="#">Entier non négatif valide</a>                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| sandbox  | <a href="#">iframe</a>                  | Règles de sécurité pour le contenu imbriqué                      | <a href="#">Ensemble non ordonné de jetons uniques séparés par des espaces</a> , <a href="#">insensibles à la casse ASCII</a> , composé de <ul style="list-style-type: none"> <li>• "<a href="#">allow-downloads</a>"</li> <li>• "<a href="#">allow-forms</a>"</li> <li>• "<a href="#">allow-modals</a>"</li> <li>• "<a href="#">allow-orientation-lock</a>"</li> <li>• "<a href="#">allow-pointer-lock</a>"</li> <li>• "<a href="#">allow-popups</a>"</li> <li>• "<a href="#">allow-popups-to-escape-sandbox</a>"</li> <li>• "<a href="#">allow-presentation</a>"</li> <li>• "<a href="#">allow-same-origin</a>"</li> <li>• "<a href="#">allow-scripts</a>"</li> <li>• "<a href="#">allow-top-navigation</a>"</li> <li>• "<a href="#">allow-top-navigation-by-user-activation</a>"</li> <li>• "<a href="#">allow-top-navigation-to-custom-protocols</a>"</li> </ul> |
| scope    | <a href="#">th</a>                      | Spécifie les cellules auxquelles la cellule d'en-tête s'applique | " <a href="#">row</a> " ; " <a href="#">col</a> " ; " <a href="#">rowgroup</a> " ; " <a href="#">colgroup</a> "                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                      |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut                | Éléments                                                                                                                                                                                                                                                                     | Description                                                       | Valeur                                                                                                                                             |
|-------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <code>selected</code>   | <a href="#">option</a>                                                                                                                                                                                                                                                       | Si l'option est sélectionnée par défaut                           | <a href="#">Attribut booléen</a>                                                                                                                   |
| <code>shape</code>      | <a href="#">area</a>                                                                                                                                                                                                                                                         | Le type de forme à créer dans une <a href="#">image cliquable</a> | " <a href="#">circle</a> "; " <a href="#">default</a> "; " <a href="#">poly</a> "; " <a href="#">rect</a> "                                        |
| <code>size</code>       | <a href="#">input</a> ; <a href="#">select</a>                                                                                                                                                                                                                               | Taille du contrôle                                                | <a href="#">Entier non négatif valide</a> supérieur à zéro                                                                                         |
| <code>sizes</code>      | <a href="#">link</a>                                                                                                                                                                                                                                                         | Tailles des icônes (pour <code>rel="icon"</code> )                | <a href="#">Ensemble non ordonné de jetons uniques séparés par des espaces</a> , <a href="#">ASCII insensible à la casse</a> , composé de tailles* |
| <code>sizes</code>      | <a href="#">img</a> ; <a href="#">source</a>                                                                                                                                                                                                                                 | Tailles d'image pour différentes mises en page                    | <a href="#">Liste de taille de source valide</a>                                                                                                   |
| <code>slot</code>       | <a href="#">Éléments HTML</a>                                                                                                                                                                                                                                                | L'emplacement souhaité de l'élément                               | <a href="#">Texte</a>                                                                                                                              |
| <code>span</code>       | <a href="#">col</a> ; <a href="#">colgroup</a>                                                                                                                                                                                                                               | Nombre de colonnes couvertes par l'élément                        | <a href="#">Entier non négatif valide</a> supérieur à zéro                                                                                         |
| <code>spellcheck</code> | <a href="#">Éléments HTML</a>                                                                                                                                                                                                                                                | Si l'élément doit faire vérifier son orthographe et sa grammaire  | "true"; "false"                                                                                                                                    |
| <code>src</code>        | <a href="#">audio</a> ; <a href="#">embed</a> ; <a href="#">iframe</a> ; <a href="#">img</a> ; <a href="#">input</a> ; <a href="#">script</a> ; <a href="#">source</a> (dans <a href="#">video</a> ou <a href="#">audio</a> ); <a href="#">track</a> ; <a href="#">video</a> | Adresse de la ressource                                           | <a href="#">URL valide non vide potentiellement entourée d'espaces</a>                                                                             |
| <code>srcdoc</code>     | <a href="#">iframe</a>                                                                                                                                                                                                                                                       | Un document à rendre dans le <a href="#">iframe</a>               | La source d'un <a href="#">un iframe srcdoc document</a> *                                                                                         |
| <code>srclang</code>    | <a href="#">track</a>                                                                                                                                                                                                                                                        | Langue de la piste de texte                                       | Balise de langue BCP 47 valide                                                                                                                     |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut | Éléments                                     | Description                                                                                                                                                                                                    | Valeur                                                                        |
|----------|----------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------|
| srcset   | <a href="#">img</a> ; <a href="#">source</a> | Images à utiliser dans différentes situations, par exemple, écrans haute résolution, petits moniteurs, etc.                                                                                                    | Liste séparée par des <a href="#">virgules des chaînes candidates d'image</a> |
| start    | <a href="#">ol</a>                           | <a href="#">Valeur de départ</a> de la liste                                                                                                                                                                   | <a href="#">Entier valide</a>                                                 |
| step     | <a href="#">input</a>                        | Granularité à faire correspondre à la valeur du contrôle de formulaire                                                                                                                                         | <a href="#">Nombre à virgule flottante valide</a> supérieur à zéro, ou "any"  |
| style    | <a href="#">Éléments HTML</a>                | Instructions de présentation et de mise en forme                                                                                                                                                               | Déclarations CSS*                                                             |
| tabindex | <a href="#">Éléments HTML</a>                | Indique si l'élément est <a href="#">focalisable</a> et <a href="#">séquentiellement focalisable</a> , et l'ordre relatif de l'élément aux fins de <a href="#">la navigation séquentielle de mise au point</a> | <a href="#">Entier valide</a>                                                 |
| target   | <a href="#">a</a> ; <a href="#">area</a>     | <a href="#">Navigable</a> pour <a href="#">la navigation par lien hypertexte</a>                                                                                                                               | <a href="#">Nom de cible navigable ou mot-clé valide</a>                      |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut  | Éléments                                     | Description                                                                                                                                | Valeur                                                   |
|-----------|----------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------|
| target    | <a href="#">base</a>                         | <a href="#">Navigable</a> par défaut pour <a href="#">la navigation par lien hypertexte</a> et <a href="#">la soumission de formulaire</a> | <a href="#">Nom de cible navigable ou mot-clé valide</a> |
| target    | <a href="#">form</a>                         | <a href="#">Navigable</a> pour <a href="#">la soumission du formulaire</a>                                                                 | <a href="#">Nom de cible navigable ou mot-clé valide</a> |
| title     | <a href="#">Éléments HTML</a>                | Informations consultatives pour l'élément                                                                                                  | <a href="#">Texte</a>                                    |
| title     | <a href="#">abbr</a> ; <a href="#">dfn</a>   | Terme complet ou extension de l'abréviation                                                                                                | <a href="#">Texte</a>                                    |
| title     | <a href="#">input</a>                        | Description du modèle (lorsqu'il est utilisé avec <a href="#">pattern</a> un attribut)                                                     | <a href="#">Texte</a>                                    |
| title     | <a href="#">link</a>                         | Titre du lien                                                                                                                              | <a href="#">Texte</a>                                    |
| title     | <a href="#">link</a> ; <a href="#">style</a> | <a href="#">Nom du jeu de feuilles de style CSS</a>                                                                                        | <a href="#">Texte</a>                                    |
| translate | <a href="#">Éléments HTML</a>                | Indique si l'élément doit être traduit lors de la localisation de la page                                                                  | " yes" ; " no"                                           |
| type      | <a href="#">a</a> ; <a href="#">link</a>     | Indice pour le type de la ressource référencée                                                                                             | <a href="#">Chaîne de type MIME valide</a>               |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut | Éléments)                                                                                                                                                                                                                    | Description                                                        | Valeur                                                                                                                                                        |
|----------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------|
| type     | <a href="#">button</a>                                                                                                                                                                                                       | Type de bouton                                                     | " <a href="#">submit</a> "; " <a href="#">reset</a> "; " <a href="#">button</a> "                                                                             |
| type     | <a href="#">embed</a> ; <a href="#">object</a> ; <a href="#">source</a>                                                                                                                                                      | Type de ressource intégrée                                         | <a href="#">Chaîne de type MIME valide</a>                                                                                                                    |
| type     | <a href="#">input</a>                                                                                                                                                                                                        | Type de contrôle de formulaire                                     | <a href="#">input</a> saisir le mot-clé                                                                                                                       |
| type     | <a href="#">ol</a>                                                                                                                                                                                                           | Type de marqueur de liste                                          | " <a href="#">1</a> "; " <a href="#">a</a> "; " <a href="#">A</a> "; " <a href="#">i</a> "; " <a href="#">I</a> "                                             |
| type     | <a href="#">script</a>                                                                                                                                                                                                       | Type de scénario                                                   | " <a href="#">module</a> "; une <a href="#">chaîne de type MIME valide</a> qui n'est pas une <a href="#">correspondance d'essence de type MIME JavaScript</a> |
| usemap   | <a href="#">img</a>                                                                                                                                                                                                          | Nom de l' <a href="#">image cliquable</a> à utiliser               | <a href="#">Référence de nom de hachage valide</a> *                                                                                                          |
| value    | <a href="#">button</a> ; <a href="#">option</a>                                                                                                                                                                              | Valeur à utiliser pour la <a href="#">soumission du formulaire</a> | <a href="#">Texte</a>                                                                                                                                         |
| value    | <a href="#">data</a>                                                                                                                                                                                                         | Valeur lisible par machine                                         | <a href="#">Texte</a> *                                                                                                                                       |
| value    | <a href="#">input</a>                                                                                                                                                                                                        | Valeur du champ de formulaire                                      | Varie*                                                                                                                                                        |
| value    | <a href="#">li</a>                                                                                                                                                                                                           | <a href="#">Valeur ordinale</a> de l'élément de liste              | <a href="#">Entier valide</a>                                                                                                                                 |
| value    | <a href="#">meter</a> ; <a href="#">progress</a>                                                                                                                                                                             | Valeur actuelle de l'élément                                       | <a href="#">Nombre à virgule flottante valide</a>                                                                                                             |
| width    | <a href="#">canvas</a> ; <a href="#">embed</a> ; <a href="#">iframe</a> ; <a href="#">img</a> ; <a href="#">input</a> ; <a href="#">object</a> ; <a href="#">source</a> (en <a href="#">picture</a> ); <a href="#">video</a> | Dimensions horizontales                                            | <a href="#">Entier non négatif valide</a>                                                                                                                     |

Liste des attributs (à l'exclusion des attributs de contenu du gestionnaire d'événements)

| Attribut | Éléments                 | Description                                                                                                       | Valeur                                              |
|----------|--------------------------|-------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------|
| wrap     | <a href="#">textarea</a> | Comment la valeur du contrôle de formulaire doit être enveloppée pour <a href="#">la soumission du formulaire</a> | " <a href="#">soft</a> " ; " <a href="#">hard</a> " |

Un astérisque (\*) dans une cellule indique que les règles réelles sont plus compliquées que celles indiquées dans le tableau ci-dessus.



Liste des attributs de contenu du gestionnaire d'événements

| Attribut      | Éléments                      | Description                                                                               | Valeur                                                           |
|---------------|-------------------------------|-------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| onauxclick    | <a href="#">Éléments HTML</a> | <a href="#">auxclick</a> gestionnaire d'événements                                        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onafterprint  | <a href="#">body</a>          | <a href="#">afterprint</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet  | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onbeforematch | <a href="#">Éléments HTML</a> | <a href="#">beforematch</a> gestionnaire d'événements                                     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onbeforeprint | <a href="#">body</a>          | <a href="#">beforeprint</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire</a>              |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut         | Éléments)                     | Description                                                                                | Valeur                                                           |
|------------------|-------------------------------|--------------------------------------------------------------------------------------------|------------------------------------------------------------------|
|                  |                               |                                                                                            | <a href="#">d'événements</a>                                     |
| onbeforeunload   | <a href="#">body</a>          | <a href="#">beforeunload</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onbeforetoggle   | <a href="#">Éléments HTML</a> | <a href="#">beforetoggle</a> gestionnaire d'événements                                     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onblur           | <a href="#">Éléments HTML</a> | <a href="#">blur</a> gestionnaire d'événements                                             | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncancel         | <a href="#">Éléments HTML</a> | <a href="#">cancel</a> gestionnaire d'événements                                           | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncanplay        | <a href="#">Éléments HTML</a> | <a href="#">canplay</a> gestionnaire d'événements                                          | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncanplaythrough | <a href="#">Éléments HTML</a> | <a href="#">canplaythrough</a> gestionnaire d'événements                                   | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onchange         | <a href="#">Éléments HTML</a> | <a href="#">change</a> gestionnaire d'événements                                           | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onclick          | <a href="#">Éléments HTML</a> | <a href="#">click</a> gestionnaire d'événements                                            | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut          | Éléments)                     | Description                                               | Valeur                                                           |
|-------------------|-------------------------------|-----------------------------------------------------------|------------------------------------------------------------------|
| onclose           | <a href="#">Éléments HTML</a> | <a href="#">close</a> gestionnaire d'événements           | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncontextlost     | <a href="#">Éléments HTML</a> | <a href="#">contextlost</a> gestionnaire d'événements     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncontextmenu     | <a href="#">Éléments HTML</a> | <a href="#">contextmenu</a> gestionnaire d'événements     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncontextrestored | <a href="#">Éléments HTML</a> | <a href="#">contextrestored</a> gestionnaire d'événements | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncopy            | <a href="#">Éléments HTML</a> | <a href="#">copy</a> gestionnaire d'événements            | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncuechange       | <a href="#">Éléments HTML</a> | <a href="#">cuechange</a> gestionnaire d'événements       | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oncut             | <a href="#">Éléments HTML</a> | <a href="#">cut</a> gestionnaire d'événements             | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondblclick        | <a href="#">Éléments HTML</a> | <a href="#">dblclick</a> gestionnaire d'événements        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondrag            | <a href="#">Éléments HTML</a> | <a href="#">drag</a> gestionnaire d'événements            | <a href="#">Attribut de contenu du gestionnaire</a>              |



Liste des attributs de contenu du gestionnaire d'événements

| Attribut         | Éléments)                     | Description                                              | Valeur                                                           |
|------------------|-------------------------------|----------------------------------------------------------|------------------------------------------------------------------|
|                  |                               |                                                          | <a href="#">d'événements</a>                                     |
| ondragend        | <a href="#">Éléments HTML</a> | <a href="#">dragend</a> gestionnaire d'événements        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondragenter      | <a href="#">Éléments HTML</a> | <a href="#">dragenter</a> gestionnaire d'événements      | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondragleave      | <a href="#">Éléments HTML</a> | <a href="#">dragleave</a> gestionnaire d'événements      | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondragover       | <a href="#">Éléments HTML</a> | <a href="#">dragover</a> gestionnaire d'événements       | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondragstart      | <a href="#">Éléments HTML</a> | <a href="#">dragstart</a> gestionnaire d'événements      | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondrop           | <a href="#">Éléments HTML</a> | <a href="#">drop</a> gestionnaire d'événements           | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ondurationchange | <a href="#">Éléments HTML</a> | <a href="#">durationchange</a> gestionnaire d'événements | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onemptied        | <a href="#">Éléments HTML</a> | <a href="#">emptied</a> gestionnaire d'événements        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut     | Éléments)                     | Description                                                                              | Valeur                                                           |
|--------------|-------------------------------|------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| onended      | <a href="#">Éléments HTML</a> | <a href="#">ended</a> gestionnaire d'événements                                          | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onerror      | <a href="#">Éléments HTML</a> | <a href="#">error</a> gestionnaire d'événements                                          | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onfocus      | <a href="#">Éléments HTML</a> | <a href="#">focus</a> gestionnaire d'événements                                          | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onformdata   | <a href="#">Éléments HTML</a> | <a href="#">formdata</a> gestionnaire d'événements                                       | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onhashchange | <a href="#">body</a>          | <a href="#">hashchange</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oninput      | <a href="#">Éléments HTML</a> | <a href="#">input</a> gestionnaire d'événements                                          | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| oninvalid    | <a href="#">Éléments HTML</a> | <a href="#">invalid</a> gestionnaire d'événements                                        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onkeydown    | <a href="#">Éléments HTML</a> | <a href="#">keydown</a> gestionnaire d'événements                                        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onkeypress   | <a href="#">Éléments HTML</a> | <a href="#">keypress</a> gestionnaire d'événements                                       | <a href="#">Attribut de contenu du gestionnaire</a>              |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut         | Éléments)                     | Description                                                                                  | Valeur                                                           |
|------------------|-------------------------------|----------------------------------------------------------------------------------------------|------------------------------------------------------------------|
|                  |                               |                                                                                              | <a href="#">d'événements</a>                                     |
| onkeyup          | <a href="#">Éléments HTML</a> | <a href="#">keyup</a> gestionnaire d'événements                                              | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onlanguagechange | <a href="#">body</a>          | <a href="#">languagechange</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onload           | <a href="#">Éléments HTML</a> | <a href="#">load</a> gestionnaire d'événements                                               | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onloadeddata     | <a href="#">Éléments HTML</a> | <a href="#">loadeddata</a> gestionnaire d'événements                                         | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onloadedmetadata | <a href="#">Éléments HTML</a> | <a href="#">loadedmetadata</a> gestionnaire d'événements                                     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onloadstart      | <a href="#">Éléments HTML</a> | <a href="#">loadstart</a> gestionnaire d'événements                                          | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmessage        | <a href="#">body</a>          | <a href="#">message</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmessageerror   | <a href="#">body</a>          | <a href="#">messageerror</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet   | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut     | Éléments)                     | Description                                                                           | Valeur                                                           |
|--------------|-------------------------------|---------------------------------------------------------------------------------------|------------------------------------------------------------------|
| onmousedown  | <a href="#">Éléments HTML</a> | <a href="#">mousedown</a> gestionnaire d'événements                                   | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmouseenter | <a href="#">Éléments HTML</a> | <a href="#">mouseenter</a> gestionnaire d'événements                                  | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmouseleave | <a href="#">Éléments HTML</a> | <a href="#">mouseleave</a> gestionnaire d'événements                                  | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmousemove  | <a href="#">Éléments HTML</a> | <a href="#">mousemove</a> gestionnaire d'événements                                   | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmouseout   | <a href="#">Éléments HTML</a> | <a href="#">mouseout</a> gestionnaire d'événements                                    | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmouseover  | <a href="#">Éléments HTML</a> | <a href="#">mouseover</a> gestionnaire d'événements                                   | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onmouseup    | <a href="#">Éléments HTML</a> | <a href="#">mouseup</a> gestionnaire d'événements                                     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onoffline    | <a href="#">body</a>          | <a href="#">offline</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ononline     | <a href="#">body</a>          | <a href="#">online</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet  | <a href="#">Attribut de contenu du gestionnaire</a>              |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut   | Éléments)                     | Description                                                                            | Valeur                                                           |
|------------|-------------------------------|----------------------------------------------------------------------------------------|------------------------------------------------------------------|
|            |                               |                                                                                        | <a href="#">d'événements</a>                                     |
| onpagehide | <a href="#">body</a>          | <a href="#">pagehide</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onpageshow | <a href="#">body</a>          | <a href="#">pageshow</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onpaste    | <a href="#">Éléments HTML</a> | <a href="#">paste</a> gestionnaire d'événements                                        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onpause    | <a href="#">Éléments HTML</a> | <a href="#">pause</a> gestionnaire d'événements                                        | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onplay     | <a href="#">Éléments HTML</a> | <a href="#">play</a> gestionnaire d'événements                                         | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onplaying  | <a href="#">Éléments HTML</a> | <a href="#">playing</a> gestionnaire d'événements                                      | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onpopstate | <a href="#">body</a>          | <a href="#">popstate</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onprogress | <a href="#">Éléments HTML</a> | <a href="#">progress</a> gestionnaire d'événements                                     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut                  | Éléments)                     | Description                                                                                    | Valeur                                                           |
|---------------------------|-------------------------------|------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| onratechange              | <a href="#">Éléments HTML</a> | <a href="#">ratechange</a> gestionnaire d'événements                                           | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onreset                   | <a href="#">Éléments HTML</a> | <a href="#">reset</a> gestionnaire d'événements                                                | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onresize                  | <a href="#">Éléments HTML</a> | <a href="#">resize</a> gestionnaire d'événements                                               | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onrejectionhandled        | <a href="#">body</a>          | <a href="#">rejectionhandled</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onscroll                  | <a href="#">Éléments HTML</a> | <a href="#">scroll</a> gestionnaire d'événements                                               | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onscrollend               | <a href="#">Éléments HTML</a> | <a href="#">scrollend</a> gestionnaire d'événements                                            | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onsecuritypolicyviolation | <a href="#">Éléments HTML</a> | <a href="#">securitypolicyviolation</a> gestionnaire d'événements                              | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onseeked                  | <a href="#">Éléments HTML</a> | <a href="#">seeked</a> gestionnaire d'événements                                               | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onseeking                 | <a href="#">Éléments HTML</a> | <a href="#">seeking</a> gestionnaire d'événements                                              | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |

Liste des attributs de contenu du gestionnaire d'événements

| Attribut     | Éléments)                     | Description                                                                           | Valeur                                                           |
|--------------|-------------------------------|---------------------------------------------------------------------------------------|------------------------------------------------------------------|
|              |                               |                                                                                       | <a href="#">d'événements</a>                                     |
| onselect     | <a href="#">Éléments HTML</a> | <a href="#">select</a> gestionnaire d'événements                                      | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onslotchange | <a href="#">Éléments HTML</a> | <a href="#">slotchange</a> gestionnaire d'événements                                  | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onstalled    | <a href="#">Éléments HTML</a> | <a href="#">stalled</a> gestionnaire d'événements                                     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onstorage    | <a href="#">body</a>          | <a href="#">storage</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onsubmit     | <a href="#">Éléments HTML</a> | <a href="#">submit</a> gestionnaire d'événements                                      | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onsuspend    | <a href="#">Éléments HTML</a> | <a href="#">suspend</a> gestionnaire d'événements                                     | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ontimeupdate | <a href="#">Éléments HTML</a> | <a href="#">timeupdate</a> gestionnaire d'événements                                  | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| ontoggle     | <a href="#">Éléments HTML</a> | <a href="#">toggle</a> gestionnaire d'événements                                      | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |

### Liste des attributs de contenu du gestionnaire d'événements

| Attribut             | Éléments)                     | Description                                                                                      | Valeur                                                           |
|----------------------|-------------------------------|--------------------------------------------------------------------------------------------------|------------------------------------------------------------------|
| onunhandledrejection | <a href="#">body</a>          | <a href="#">unhandledrejection</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onunload             | <a href="#">body</a>          | <a href="#">unload</a> gestionnaire d'événements pour <a href="#">Window</a> l'objet             | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onvolumechange       | <a href="#">Éléments HTML</a> | <a href="#">volumechange</a> gestionnaire d'événements                                           | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onwaiting            | <a href="#">Éléments HTML</a> | <a href="#">waiting</a> gestionnaire d'événements                                                | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |
| onwheel              | <a href="#">Éléments HTML</a> | <a href="#">wheel</a> gestionnaire d'événements                                                  | <a href="#">Attribut de contenu du gestionnaire d'événements</a> |

## Interfaces d'éléments

*Cette section est non normative.*

### Liste des interfaces pour les éléments

| Éléments)               | Interface(s)                                                                                      |
|-------------------------|---------------------------------------------------------------------------------------------------|
| <a href="#">a</a>       | <a href="#">HTMLAnchorElement</a> : <a href="#">HTMLElement</a>                                   |
| <a href="#">abbr</a>    | <a href="#">HTMLElement</a>                                                                       |
| <a href="#">address</a> | <a href="#">HTMLElement</a>                                                                       |
| <a href="#">area</a>    | <a href="#">HTMLAreaElement</a> : <a href="#">HTMLElement</a>                                     |
| <a href="#">article</a> | <a href="#">HTMLElement</a>                                                                       |
| <a href="#">aside</a>   | <a href="#">HTMLElement</a>                                                                       |
| <a href="#">audio</a>   | <a href="#">HTMLAudioElement</a> : <a href="#">HTMLMediaElement</a> : <a href="#">HTMLElement</a> |



## Liste des interfaces pour les éléments

| Éléments)                         | Interface(s)                                               |
|-----------------------------------|------------------------------------------------------------|
| <a href="#"><u>b</u></a>          | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>base</u></a>       | <a href="#"><u>HTMLBaseElement:HTMLElement</u></a>         |
| <a href="#"><u>bdi</u></a>        | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>bdo</u></a>        | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>blockquote</u></a> | <a href="#"><u>HTMLQuoteElement:HTMLElement</u></a>        |
| <a href="#"><u>body</u></a>       | <a href="#"><u>HTMLBodyElement:HTMLElement</u></a>         |
| <a href="#"><u>br</u></a>         | <a href="#"><u>HTMLBRElement:HTMLElement</u></a>           |
| <a href="#"><u>button</u></a>     | <a href="#"><u>HTMLButtonElement:HTMLElement</u></a>       |
| <a href="#"><u>canvas</u></a>     | <a href="#"><u>HTMLCanvasElement:HTMLElement</u></a>       |
| <a href="#"><u>caption</u></a>    | <a href="#"><u>HTMLTableCaptionElement:HTMLElement</u></a> |
| <a href="#"><u>cite</u></a>       | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>code</u></a>       | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>col</u></a>        | <a href="#"><u>HTMLTableColElement:HTMLElement</u></a>     |
| <a href="#"><u>colgroup</u></a>   | <a href="#"><u>HTMLTableColElement:HTMLElement</u></a>     |
| <a href="#"><u>data</u></a>       | <a href="#"><u>HTMLDataElement:HTMLElement</u></a>         |
| <a href="#"><u>datalist</u></a>   | <a href="#"><u>HTMLDataListElement:HTMLElement</u></a>     |
| <a href="#"><u>dd</u></a>         | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>del</u></a>        | <a href="#"><u>HTMLModElement:HTMLElement</u></a>          |
| <a href="#"><u>details</u></a>    | <a href="#"><u>HTMLDetailsElement:HTMLElement</u></a>      |
| <a href="#"><u>dfn</u></a>        | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>dialog</u></a>     | <a href="#"><u>HTMLDialogElement:HTMLElement</u></a>       |
| <a href="#"><u>div</u></a>        | <a href="#"><u>HTMLDivElement:HTMLElement</u></a>          |
| <a href="#"><u>dl</u></a>         | <a href="#"><u>HTMLDListElement:HTMLElement</u></a>        |
| <a href="#"><u>dt</u></a>         | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>em</u></a>         | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>embed</u></a>      | <a href="#"><u>HTMLEmbedElement:HTMLElement</u></a>        |
| <a href="#"><u>fieldset</u></a>   | <a href="#"><u>HTMLFieldSetElement:HTMLElement</u></a>     |
| <a href="#"><u>figcaption</u></a> | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>figure</u></a>     | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>footer</u></a>     | <a href="#"><u>HTMLElement</u></a>                         |
| <a href="#"><u>form</u></a>       | <a href="#"><u>HTMLFormElement:HTMLElement</u></a>         |
| <a href="#"><u>h1</u></a>         | <a href="#"><u>HTMLHeadingElement:HTMLElement</u></a>      |
| <a href="#"><u>h2</u></a>         | <a href="#"><u>HTMLHeadingElement:HTMLElement</u></a>      |
| <a href="#"><u>h3</u></a>         | <a href="#"><u>HTMLHeadingElement:HTMLElement</u></a>      |
| <a href="#"><u>h4</u></a>         | <a href="#"><u>HTMLHeadingElement:HTMLElement</u></a>      |
| <a href="#"><u>h5</u></a>         | <a href="#"><u>HTMLHeadingElement:HTMLElement</u></a>      |

## Liste des interfaces pour les éléments

| Éléments)                       | Interface(s)                                            |
|---------------------------------|---------------------------------------------------------|
| <a href="#"><u>h6</u></a>       | <a href="#"><u>HTMLHeadingElement:HTMLElement</u></a>   |
| <a href="#"><u>head</u></a>     | <a href="#"><u>HTMLHeadElement:HTMLElement</u></a>      |
| <a href="#"><u>header</u></a>   | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>hgroup</u></a>   | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>hr</u></a>       | <a href="#"><u>HTMLHRElement:HTMLElement</u></a>        |
| <a href="#"><u>html</u></a>     | <a href="#"><u>HTMLHtmlElement:HTMLElement</u></a>      |
| <a href="#"><u>i</u></a>        | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>iframe</u></a>   | <a href="#"><u>HTMLIFrameElement:HTMLElement</u></a>    |
| <a href="#"><u>img</u></a>      | <a href="#"><u>HTMLImageElement:HTMLElement</u></a>     |
| <a href="#"><u>input</u></a>    | <a href="#"><u>HTMLInputElement:HTMLElement</u></a>     |
| <a href="#"><u>ins</u></a>      | <a href="#"><u>HTMLModElement:HTMLElement</u></a>       |
| <a href="#"><u>kbd</u></a>      | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>label</u></a>    | <a href="#"><u>HTMLLabelElement:HTMLElement</u></a>     |
| <a href="#"><u>legend</u></a>   | <a href="#"><u>HTMLLegendElement:HTMLElement</u></a>    |
| <a href="#"><u>li</u></a>       | <a href="#"><u>HTMLLIElement:HTMLElement</u></a>        |
| <a href="#"><u>link</u></a>     | <a href="#"><u>HTMLLinkElement:HTMLElement</u></a>      |
| <a href="#"><u>main</u></a>     | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>map</u></a>      | <a href="#"><u>HTMLMapElement:HTMLElement</u></a>       |
| <a href="#"><u>mark</u></a>     | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>menu</u></a>     | <a href="#"><u>HTMLMenuElement:HTMLElement</u></a>      |
| <a href="#"><u>meta</u></a>     | <a href="#"><u>HTMLMetaElement:HTMLElement</u></a>      |
| <a href="#"><u>meter</u></a>    | <a href="#"><u>HTMLMeterElement:HTMLElement</u></a>     |
| <a href="#"><u>nav</u></a>      | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>noscript</u></a> | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>object</u></a>   | <a href="#"><u>HTMLObjectElement:HTMLElement</u></a>    |
| <a href="#"><u>ol</u></a>       | <a href="#"><u>HTMLOLListElement:HTMLElement</u></a>    |
| <a href="#"><u>optgroup</u></a> | <a href="#"><u>HTMLOptGroupElement:HTMLElement</u></a>  |
| <a href="#"><u>option</u></a>   | <a href="#"><u>HTMLOptionElement:HTMLElement</u></a>    |
| <a href="#"><u>output</u></a>   | <a href="#"><u>HTMLOutputElement:HTMLElement</u></a>    |
| <a href="#"><u>p</u></a>        | <a href="#"><u>HTMLParagraphElement:HTMLElement</u></a> |
| <a href="#"><u>picture</u></a>  | <a href="#"><u>HTMLPictureElement:HTMLElement</u></a>   |
| <a href="#"><u>pre</u></a>      | <a href="#"><u>HTMLPreElement:HTMLElement</u></a>       |
| <a href="#"><u>progress</u></a> | <a href="#"><u>HTMLProgressElement:HTMLElement</u></a>  |
| <a href="#"><u>q</u></a>        | <a href="#"><u>HTMLQuoteElement:HTMLElement</u></a>     |
| <a href="#"><u>rp</u></a>       | <a href="#"><u>HTMLElement</u></a>                      |
| <a href="#"><u>rt</u></a>       | <a href="#"><u>HTMLElement</u></a>                      |

## Liste des interfaces pour les éléments

| Éléments)                              | Interface(s)                                                              |
|----------------------------------------|---------------------------------------------------------------------------|
| <a href="#">ruby</a>                   | <a href="#">HTMLElement</a>                                               |
| <a href="#">s</a>                      | <a href="#">HTMLElement</a>                                               |
| <a href="#">samp</a>                   | <a href="#">HTMLElement</a>                                               |
| <a href="#">script</a>                 | <a href="#">HTMLScriptElement:HTMLElement</a>                             |
| <a href="#">section</a>                | <a href="#">HTMLElement</a>                                               |
| <a href="#">select</a>                 | <a href="#">HTMLSelectElement:HTMLElement</a>                             |
| <a href="#">slot</a>                   | <a href="#">HTMLSlotElement:HTMLElement</a>                               |
| <a href="#">small</a>                  | <a href="#">HTMLElement</a>                                               |
| <a href="#">source</a>                 | <a href="#">HTMLSourceElement:HTMLElement</a>                             |
| <a href="#">span</a>                   | <a href="#">HTMLSpanElement:HTMLElement</a>                               |
| <a href="#">strong</a>                 | <a href="#">HTMLElement</a>                                               |
| <a href="#">style</a>                  | <a href="#">HTMLStyleElement:HTMLElement</a>                              |
| <a href="#">sub</a>                    | <a href="#">HTMLElement</a>                                               |
| <a href="#">summary</a>                | <a href="#">HTMLElement</a>                                               |
| <a href="#">sup</a>                    | <a href="#">HTMLElement</a>                                               |
| <a href="#">table</a>                  | <a href="#">HTMLTableElement:HTMLElement</a>                              |
| <a href="#">tbody</a>                  | <a href="#">HTMLTableSectionElement:HTMLElement</a>                       |
| <a href="#">td</a>                     | <a href="#">HTMLTableCellElement:HTMLElement</a>                          |
| <a href="#">template</a>               | <a href="#">HTMLTemplateElement:HTMLElement</a>                           |
| <a href="#">textarea</a>               | <a href="#">HTMLTextAreaElement:HTMLElement</a>                           |
| <a href="#">tfoot</a>                  | <a href="#">HTMLTableSectionElement:HTMLElement</a>                       |
| <a href="#">th</a>                     | <a href="#">HTMLTableCellElement:HTMLElement</a>                          |
| <a href="#">thead</a>                  | <a href="#">HTMLTableSectionElement:HTMLElement</a>                       |
| <a href="#">time</a>                   | <a href="#">HTMLTimeElement:HTMLElement</a>                               |
| <a href="#">title</a>                  | <a href="#">HTMLTitleElement:HTMLElement</a>                              |
| <a href="#">tr</a>                     | <a href="#">HTMLTableRowElement:HTMLElement</a>                           |
| <a href="#">track</a>                  | <a href="#">HTMLTrackElement:HTMLElement</a>                              |
| <a href="#">u</a>                      | <a href="#">HTMLElement</a>                                               |
| <a href="#">ul</a>                     | <a href="#">HTMLUListElement:HTMLElement</a>                              |
| <a href="#">var</a>                    | <a href="#">HTMLElement</a>                                               |
| <a href="#">video</a>                  | <a href="#">HTMLVideoElement: HTMLMediaElement:HTMLElement</a>            |
| <a href="#">wbr</a>                    | <a href="#">HTMLElement</a>                                               |
| <a href="#">éléments personnalisés</a> | fourni par l'auteur de l'élément (hérite de <a href="#">HTMLElement</a> ) |

# Toutes les interfaces

*Cette section est non normative.*

- [AudioTrack](#)
- [AudioTrackList](#)
- [BarProp](#)
- [BeforeUnloadEvent](#)
- [BroadcastChannel](#)
- [CanvasGradient](#)
- [CanvasPattern](#)
- [CanvasRenderingContext2D](#)
- [CustomElementRegistry](#)
- [DOMParser](#)
- [DOMStringList](#)
- [DOMStringMap](#)
- [DataTransfer](#)
- [DataTransferItem](#)
- [DataTransferItemList](#)
- [DedicatedWorkerGlobalScope](#)
- [Document](#), [partiel 1 1](#)
- [DragEvent](#)
- [ElementInternals](#)
- [ErrorEvent](#)
- [EventSource](#)
- [External](#)
- [FormDataEvent](#)
- [HTMLAllCollection](#)
- [HTMLAnchorElement](#), [partiel](#)
- [HTMLAreaElement](#), [partiel](#)
- [HTMLAudioElement](#)
- [HTMLBRElement](#), [partiel](#)
- [HTMLBaseElement](#)
- [HTMLBodyElement](#), [partiel](#)
- [HTMLButtonElement](#)
- [HTMLCanvasElement](#)
- [HTMLDListElement](#), [partiel](#)
- [HTMLDataElement](#)
- [HTMLDataListElement](#)
- [HTMLDetailsElement](#)
- [HTMLDialogElement](#)
- [HTMLDirectoryElement](#)
- [HTMLDivElement](#), [partiel](#)
- [HTMLElement](#)
- [HTMLEmbedElement](#), [partiel](#)
- [HTMLFieldSetElement](#)
- [HTMLFontElement](#)
- [HTMLFormControlsCollection](#)
- [HTMLFormElement](#)
- [HTMLFrameElement](#)
- [HTMLFrameSetElement](#)
- [HTMLHRElement](#), [partiel](#)

- [HTMLHeadElement](#)
- [HTMLHeadingElement](#), [partiel](#)
- [HTMLHtmlElement](#), [partiel](#)
- [HTMLIFrameElement](#), [partiel](#)
- [HTMLImageElement](#), [partiel](#)
- [HTMLInputElement](#), [partiel](#)
- [HTMLLIElement](#), [partiel](#)
- [HTMLLabelElement](#)
- [HTMLLegendElement](#), [partiel](#)
- [HTMMLinkElement](#), [partiel](#)
- [HTMLMapElement](#)
- [HTMLMarqueeElement](#)
- [HTMLMediaElement](#)
- [HTMLMenuElement](#), [partiel](#)
- [HTMLMetaElement](#), [partiel](#)
- [HTMLMeterElement](#)
- [HTMLModElement](#)
- [HTMLOListElement](#), [partiel](#)
- [HTMLObjectElement](#), [partiel](#)
- [HTMLOptGroupElement](#)
- [HTMLOptionElement](#)
- [HTMLOptionsCollection](#)
- [HTMLOutputElement](#)
- [HTMLParagraphElement](#), [partiel](#)
- [HTMLParamElement](#)
- [HTMLPictureElement](#)
- [HTMLPreElement](#), [partiel](#)
- [HTMLProgressElement](#)
- [HTMLQuoteElement](#)
- [HTMLScriptElement](#), [partiel](#)
- [HTMLSelectElement](#)
- [HTMLSlotElement](#)
- [HTMLSourceElement](#)
- [HTMLSpanElement](#)
- [HTMLStyleElement](#), [partiel](#)
- [HTMLTableCaptionElement](#), [partiel](#)
- [HTMLTableCellElement](#), [partiel](#)
- [HTMLTableColElement](#), [partiel](#)
- [HTMLTableElement](#), [partiel](#)
- [HTMLTableRowElement](#), [partiel](#)
- [HTMLTableSectionElement](#), [partiel](#)
- [HTMLTemplateElement](#)
- [HTMLTextAreaElement](#)
- [HTMLTimeElement](#)
- [HTMLTitleElement](#)
- [HTMLTrackElement](#)
- [HTMLULListElement](#), [partiel](#)
- [HTMLUnknownElement](#)
- [HTMLVideoElement](#)
- [HashChangeEvent](#)
- [History](#)

- [ImageBitmap](#)
- [ImageBitmapRenderingContext](#)
- [ImageData](#)
- [Location](#)
- [MediaError](#)
- [MessageChannel](#)
- [MessageEvent](#)
- [MessagePort](#)
- [MimeType](#)
- [MimeTypeArray](#)
- [Navigator](#), [partiel](#)
- [OffscreenCanvas](#)
- [OffscreenCanvasRenderingContext2D](#)
- [PageTransitionEvent](#)
- [Path2D](#)
- [Plugin](#)
- [PluginArray](#)
- [PopStateEvent](#)
- [PromiseRejectionEvent](#)
- [RadioNodeList](#)
- [SharedWorker](#)
- [SharedWorkerGlobalScope](#)
- [Storage](#)
- [StorageEvent](#)
- [SubmitEvent](#)
- [TextMetrics](#)
- [TextTrack](#)
- [TextTrackCue](#)
- [TextTrackCueList](#)
- [TextTrackList](#)
- [TimeRanges](#)
- [ToggleEvent](#)
- [TrackEvent](#)
- [UserActivation](#)
- [ValidityState](#)
- [VideoTrack](#)
- [VideoTrackList](#)
- [Window](#), [partiel](#)
- [Worker](#)
- [WorkerGlobalScope](#)
- [WorkerLocation](#)
- [WorkerNavigator](#)
- [Worklet](#)
- [WorkletGlobalScope](#)

## Événements

*Cette section est non normative.*

Le tableau suivant répertorie les événements déclenchés par ce document, à l'exclusion de ceux déjà définis dans [les événements d'élément multimédia](#) et [les événements de glisser-déposer](#).

Liste des événements













| Événement                            | Interface                         | Des cibles intéressantes                                        | Description                                                                                                                                        |
|--------------------------------------|-----------------------------------|-----------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------|
| <b>DOMContentLoaded</b><br>✓ MD<br>N | Event                             | <a href="#">Document</a>                                        | Lancé à la <a href="#">Document</a> fois que l'analyseur a terminé                                                                                 |
| <b>afterprint</b><br>✓ MD<br>N       | Event                             | <a href="#">Window</a>                                          | Tiré <a href="#">Window</a> après l'impression                                                                                                     |
| <b>beforeprint</b><br>✓ MD<br>N      | Event                             | <a href="#">Window</a>                                          | Tiré à l' <a href="#">Window</a> avant impression                                                                                                  |
| <b>beforematch</b><br>Δ MD<br>N      | Event                             | Éléments                                                        | Tiré sur les éléments avec l' <a href="#">hidden=until-found</a> attribut avant qu'ils ne soient révélés.                                          |
| <b>beforetoggle</b>                  | <a href="#">ToggleEvent</a>       | Éléments                                                        | Lancé sur les éléments avec l' <a href="#">popover</a> attribut lors de leur transition entre l'affichage et le masquage                           |
| <b>beforeunload</b><br>✓ MD<br>N     | <a href="#">BeforeUnloadEvent</a> | <a href="#">Window</a>                                          | Lancé lorsque <a href="#">Window</a> la page est sur le point d'être déchargée, au cas où la page souhaiterait afficher une invite d'avertissement |
| <b>blur</b>                          | Event                             | <a href="#">Window</a> , éléments                               | Tiré sur les nœuds lorsqu'ils cessent d'être <a href="#">ciblés</a>                                                                                |
| <b>cancel</b><br>✓ MD<br>N           | Event                             | <a href="#">dialog</a> éléments, <a href="#">input</a> éléments | Déclenché sur <a href="#">dialog</a> des éléments lorsqu'ils sont annulés par                                                                      |

# Liste des événements

| Événement                       | Interface                     | Des cibles intéressantes                                                | Description                                                                                                                                                                                                                                                          |
|---------------------------------|-------------------------------|-------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                 |                               |                                                                         | l'utilisateur (par exemple, en appuyant sur la touche Échap) ou sur <a href="#">input</a> des éléments à l'état <a href="#">Fichier</a> lorsque l'utilisateur ne modifie pas sa sélection                                                                            |
| <b>change</b><br>✓ MD<br>N      | <a href="#">Event</a>         | Champs de formulaire                                                    | Lancé sur les contrôles lorsque l'utilisateur valide un changement de valeur (voir aussi l' <a href="#">input</a> événement)                                                                                                                                         |
| <a href="#">click</a>           | <a href="#">Pointer Event</a> | Éléments                                                                | Normalement un événement de souris ; également tiré synthétiquement sur un élément avant que son <a href="#">comportement d'activation</a> ne soit exécuté, lorsqu'un élément est activé à partir d'un périphérique d'entrée sans pointeur (par exemple, un clavier) |
| <b>close</b><br>✓ MD<br>N       | <a href="#">Event</a>         | <a href="#">dialog</a> éléments                                         | Tiré sur <a href="#">dialog</a> des éléments lorsqu'ils sont fermés                                                                                                                                                                                                  |
| <b>connect</b><br>✓ MD<br>N     | <a href="#">Message Event</a> | <a href="#">SharedWorkerGlobalScope</a>                                 | Déclenché sur la portée globale d'un nœud de calcul partagé lorsqu'un nouveau client se connecte                                                                                                                                                                     |
| <b>contextlost</b><br>⚠ MD<br>N | <a href="#">Event</a>         | <a href="#">canvas</a> éléments, <a href="#">OffscreenCanvas</a> objets | Déclenché lorsque le <a href="#">CanvasRenderingContext2D</a> ou correspondant <a href="#">Offsc</a>                                                                                                                                                                 |



# Liste des événements

| Événement                                                                                                                                                                                                                                                                           | Interface                                           | Des cibles intéressantes                                                                 | Description                                                                                                                                                                                                          |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------|------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                                                                                                                                                                                                                                                                     |                                                     |                                                                                          | <a href="#">reenCanvasRenderingContext2D</a> est perdu                                                                                                                                                               |
| <b>contextlost</b><br> <br>      | <a href="#">Event</a>                               | <a href="#">canvas</a> éléments, <a href="#">OffscreenCanvas</a> objets                  | Déclenché lorsque le <a href="#">CanvasRenderingContext2D</a> ou correspondant <a href="#">OffscreenCanvasRenderingContext2D</a> est restauré après avoir été perdu                                                  |
| <b>error</b><br> <br>            | <a href="#">Event</a> ou <a href="#">ErrorEvent</a> | Objets de portée globale, <a href="#">Worker</a> objets, éléments, objets liés au réseau | Lancé lorsque des erreurs inattendues se produisent (par exemple, des erreurs de réseau, des erreurs de script, des erreurs de décodage)                                                                             |
| <b>focus</b>                                                                                                                                                                                                                                                                        | <a href="#">Event</a>                               | <a href="#">Window</a> , éléments                                                        | Tiré sur les nœuds <a href="#">qui gagnent en concentration</a>                                                                                                                                                      |
| <b>formdata</b><br> <br>   | <a href="#">FormDataEvent</a>                       | <a href="#">form</a> éléments                                                            | Tiré sur un <a href="#">form</a> élément lors de <a href="#">la construction de la liste d'entrées</a>                                                                                                               |
| <b>hashchange</b><br> <br> | <a href="#">HashChangeEvent</a>                     | <a href="#">Window</a>                                                                   | Déclenché lorsque <a href="#">Window</a> la partie <a href="#">fragment de l'URL</a> du document change                                                                                                              |
| <a href="#">input</a>                                                                                                                                                                                                                                                               | <a href="#">Event</a>                               | Éléments                                                                                 | Lancé lorsque l'utilisateur modifie le <a href="#">contenteditable</a> contenu de l'élément ou la valeur du contrôle de formulaire. Voir aussi l' <a href="#">change</a> événement pour les contrôles de formulaire. |
| <b>invalid</b>                                                                                                                                                                                                                                                                      | <a href="#">Event</a>                               | Champs de formulaire                                                                     | Lancé sur les contrôles lors de la validation du                                                                                                                                                                     |

# Liste des événements

| Événement                                              | Interface        | Des cibles intéressantes                                                                                                                                              | Description                                                                                                                                                                                 |
|--------------------------------------------------------|------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <div>✓ MD</div> <div>N</div>                           |                  |                                                                                                                                                                       | formulaire s'ils ne satisfont pas leurs contraintes                                                                                                                                         |
| <div>languagechange</div> <div>✓ MD</div> <div>N</div> | Event            | Objets de portée globale                                                                                                                                              | Lancé sur l'objet de portée globale lorsque les langues préférées de l'utilisateur changent                                                                                                 |
| <div>load</div>                                        | Event            | <u>Window</u> , éléments                                                                                                                                              | Déclenché à la <u>Window</u> fin du chargement du document ; tiré sur un élément contenant une ressource (par exemple <u>img</u> , <u>embed</u> ) lorsque sa ressource a fini de se charger |
| <div>message</div> <div>✓ MD</div> <div>N</div>        | Message<br>Event | <u>Window</u> , <u>EventSource</u> , <u>MessagePort</u> , <u>BroadcastChannel</u> , <u>DedicatedWorkerGlobalScope</u> , <u>Worker</u> , <u>ServiceWorkerContainer</u> | Tiré sur un objet lorsqu'il reçoit un message                                                                                                                                               |
| <div>messageerror</div> <div>✓ MD</div> <div>N</div>   | Message<br>Event | <u>Window</u> , <u>MessagePort</u> , <u>BroadcastChannel</u> , <u>DedicatedWorkerGlobalScope</u> , <u>Worker</u> , <u>ServiceWorkerContainer</u>                      | Lancé sur un objet lorsqu'il reçoit un message qui ne peut pas être désérialisé                                                                                                             |
| <div>offline</div> <div>✓ MD</div> <div>N</div>        | Event            | Objets de portée globale                                                                                                                                              | Lancé sur l'objet de portée globale lorsque les connexions réseau échouent                                                                                                                  |
| <div>online</div> <div>✓ MD</div> <div>N</div>         | Event            | Objets de portée globale                                                                                                                                              | Lancé sur l'objet de portée globale lorsque les connexions réseau reviennent                                                                                                                |
| <div>open</div> <div>✓ MD</div> <div>N</div>           | Event            | <u>EventSource</u>                                                                                                                                                    | Tiré sur <u>EventSource</u> des objets lorsqu'une connexion est établie                                                                                                                     |

# Liste des événements

| Événement                                    | Interface                    | Des cibles intéressantes                     | Description                                                                                                                              |
|----------------------------------------------|------------------------------|----------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------|
| <p>pagehide</p> <p>✓ MD</p> <p>N</p>         | <p>PageTransitionEvent</p>   | <p>Window</p>                                | <p>Déclenché lorsque <a href="#">l'entrée d'historique de sessionWindow</a> de la page cesse d'être l' <a href="#">entrée active</a></p> |
| <p>pageshow</p> <p>✓ MD</p> <p>N</p>         | <p>PageTransitionEvent</p>   | <p>Window</p>                                | <p>Déclenché lorsque <a href="#">l'entrée d'historique de sessionWindow</a> de la page devient l' <a href="#">entrée active</a></p>      |
| <p>pointerdown</p>                           | <p>PointerEvent</p>          | <p>Éléments et <a href="#">Text</a>nœuds</p> | <p>Lancé sur le <a href="#">nœud source</a> lorsque l'utilisateur tente de lancer une opération de glisser-déposer</p>                   |
| <p>popstate</p> <p>✓ MD</p> <p>N</p>         | <p>PopStateEvent</p>         | <p>Window</p>                                | <p>Lancé au <a href="#">Window</a>moment où, dans certains cas, <a href="#">la traversée de l'historique de session</a></p>              |
| <p>readystatechange</p> <p>✓ MD</p> <p>N</p> | <p>Event</p>                 | <p>Document</p>                              | <p>Lancé à la <a href="#">Document</a>fin de l'analyse et à nouveau lorsque toutes ses sous-ressources ont fini de se charger</p>        |
| <p>rejectionhandled</p>                      | <p>PromiseRejectionEvent</p> | <p>Objets de portée globale</p>              | <p>Lancé sur des objets de portée globale lorsqu'un rejet de promesse précédemment non géré est traité</p>                               |
| <p>reset</p> <p>✓ MD</p> <p>N</p>            | <p>Event</p>                 | <p><a href="#">form</a>éléments</p>          | <p>Tiré sur un <a href="#">form</a>élément lorsqu'il est <a href="#">réinitialisé</a></p>                                                |
| <p>select</p> <p>✓ MD</p> <p>N</p>           | <p>Event</p>                 | <p>Champs de formulaire</p>                  | <p>Lancé sur les contrôles de formulaire lorsque leur sélection de</p>                                                                   |

# Liste des événements

| Événement                       | Interface             | Des cibles intéressantes                  | Description                                                                                                                                                                                   |
|---------------------------------|-----------------------|-------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
|                                 |                       |                                           | texte est ajustée (que ce soit par une API ou par l'utilisateur)                                                                                                                              |
| storage<br>✓ MD<br>N            | StorageEvent          | Window                                    | Déclenché lors <u>Window</u> d'un événement lorsque les zones correspondantes <u>localStorage</u> ou de stockage changent <u>sessionStorage</u>                                               |
| submit<br>✓ MD<br>N             | SubmitEvent           | <u>form</u> éléments                      | Tiré sur un <u>form</u> élément lorsqu'il est <u>soumis</u>                                                                                                                                   |
| toggle<br>✓ MD<br>N             | Event ou ToggleEvent  | <u>details</u> et éléments <u>popover</u> | Tiré sur <u>details</u> des éléments lorsqu'ils s'ouvrent ou se ferment ; déclenché sur les éléments avec l' <u>popover</u> attribut lors de leur transition entre l'affichage et le masquage |
| unhandledrejection<br>✓ MD<br>N | PromiseRejectionEvent | Objets de portée globale                  | Lancé sur des objets de portée globale lorsqu'un rejet de promesse n'est pas géré                                                                                                             |
| unload<br>✓ MD<br>N             | Event                 | Window                                    | Tiré sur l' <u>Window</u> objet lorsque la page disparaît                                                                                                                                     |
| visibilitychange<br>✓ MD<br>N   | Event                 | Document                                  | Déclenché sur l' <u>Document</u> objet lorsque la page devient visible ou masquée pour l'utilisateur                                                                                          |

## En-têtes HTTP

*Cette section est non normative.*

Les en-têtes de requête HTTP suivants sont définis par cette spécification :

- ``Last-Event-ID``
- ``Ping-From``
- ``Ping-To``

Les en-têtes de réponse HTTP suivants sont définis par cette spécification :

- ``Cross-Origin-Embedder-Policy``
- ``Cross-Origin-Embedder-Policy-Report-Only``
- ``Cross-Origin-Opener-Policy``
- ``Cross-Origin-Opener-Policy-Report-Only``
- ``Origin-Agent-Cluster``
- ``Refresh``
- ``X-Frame-Options``

## types MIME

*Cette section est non normative.*

Les types MIME suivants sont mentionnés dans cette spécification :

`application/atom+xml`

Atome [\[ATOME\]](#)

`application/json`

JSON [\[JSON\]](#)

`application/octet-stream`

Données binaires génériques [\[RFC2046\]](#)

`application/microdata+json`

Microdonnées au format JSON

`application/rss+xml`

RSS

application/x-www-form-urlencoded

Soumission du formulaire

application/xhtml+xml

HTML

application/xml

XML [\[XML\]](#) [\[RFC7303\]](#)

image/gif

Images GIF [\[GIF\]](#)

image/jpeg

Images JPEG [\[JPEG\]](#)

image/png

Images PNG [\[PNG\]](#)

image/svg+xml

Images SVG [\[SVG\]](#)

multipart/form-data

Soumission de formulaire [\[RFC7578\]](#)

multipart/mixed

Contenu mixte générique [\[RFC2046\]](#)

multipart/x-mixed-replace

Poussée du serveur de streaming

text/css

CSS [\[CSS\]](#)

text/event-stream

Flux d'événements envoyés par le serveur

text/javascript

JavaScript [\[JAVASCRIPT\]](#) [\[RFC9239\]](#)

text/json

JSON (type hérité)

text/plain

Texte brut générique [\[RFC2046\]](#) [\[RFC3676\]](#)

text/html

HTML

text/ping

Audit des liens hypertexte

text/uri-list

Liste des URL [\[RFC2483\]](#)

text/vcard

vCard [\[RFC6350\]](#)

text/vtt

WebVTT [\[WEBVTT\]](#)

text/xml

XML [\[XML\]](#) [\[RFC7303\]](#)

video/mp4

Vidéo MPEG-4 [\[RFC4337\]](#)

video/mpeg

Vidéo MPEG [\[RFC2046\]](#)

## 1. [Les références](#)

# Les références

Toutes les références sont normatives sauf si elles sont marquées "Non normatif".

## [ABNF]

[BNF augmenté pour les spécifications de syntaxe : ABNF](#) , D. Crocker, P. Overell. IETF.

## [À PROPOS DE]

[Le schéma d'URI "à propos"](#) , S. Moonesamy. IETF.

## [APNG]

(Non normatif) [Spécification APNG](#) . S. Parmenter, V. Vukicevic, A. Smith. Mozilla.

## [ARIA]

[Applications Internet riches accessibles \(WAI-ARIA\)](#) , J. Diggs, J. Nurthen, M. Cooper. W3C.

## [ARIAHTML]

[ARIA en HTML](#) , S. Faulkner, S. O'Hara. W3C.

## [UNE ÉTIQUETTE]

(Non normatif) [Authoring Tool Accessibility Guidelines \(ATAG\) 2.0](#) , J. Richards, J. Spellman, J. Treviranus. W3C.

## [ATOME]

(Non normatif) [Le format de syndication Atom](#) , M. Nottingham, R. Sayre. IETF.

## [BATTERIE]

(Non normatif) [API Battery Status](#) , A. Kostiainen, M. Lamouri. W3C.

**[BCP47]**

[Balises pour identifier les langues ; Correspondance des étiquettes de langue](#) , A. Phillips, M. Davis. IETF.

**[BEZIER]**

*Courbes à pôles* , P. de Casteljaou. INPI, 1959.

**[BIDI]**

[UAX #9 : Algorithme bidirectionnel Unicode](#) , M. Davis. Consortium Unicode.

**[BOCU1]**

(Non normatif) [UTN #6 : BOCU-1 : Compression Unicode compatible MIME](#) , M. Scherer, M. Davis. Consortium Unicode.

**[CESU8]**

(Non normatif) [UTR #26 : Schéma de codage de compatibilité pour UTF-16 : 8-BIT \(CESU-8\)](#) , T. Phipps. Consortium Unicode.

**[CHAMOD]**

Modèle de caractères (non normatif) [pour le World Wide Web 1.0 : Fondamentaux](#) , M. Dürst, F. Yergeau, R. Ishida, M. Wolf, T. Texin. W3C.

**[NORME CHARMOD]**

Modèle de caractères (non normatif) [pour le World Wide Web : Correspondance de chaînes](#) , A. Phillips. W3C.

**[CLIPBOARD-APIS]**

[API Presse-papiers et événements](#) , G. Kacmarcik, A. Snigdha. W3C.

**[COMPOSITE]**

[Compositing et Blending](#) , R. Cabanier, N. Andronikos. W3C.

**[CALCUTABLE]**

(Non normatif) [Sur les nombres calculables, avec une application à l'Entscheidungsproblem](#) , A. Turing. Dans *Actes de la London Mathematical Society* , série 2, volume 42, pages 230-265. Société mathématique de Londres, 1937.

**[CONSOLE]**

[Console](#) , T. Stock, R. Kowalski, D. Farolino. WHATWG.

**[BISCUITS]**

[Mécanisme de gestion d'état HTTP](#) , A. Barth. IETF.

**[CREDMAN]**

[Gestion des accréditations](#) , N. Satragno, J. Hodges, M. West. W3C.

**[CSP]**

[Politique de sécurité du contenu](#) , M. West, D. Veditz. W3C.

**[CSS]**



[Feuilles de style en cascade Niveau 2 Révision 2](#) , B. Bos, T. Çelik, I. Hickson, H. Lie. W3C.

**[CSSALIGN]**

[CSS Box Alignment](#) , E. Etemad, T. Atkins. W3C.

**[CSSANIMATIONS]**

[CSS Animations](#) , D. Jackson, D. Hyatt, C. Marrin, S. Galineau, L. Baron. W3C.

**[CSSATTR]**

[Attributs de style CSS](#) , T. Çelik, E. Etemad. W3C.

**[CSSBG]**

[Arrière-plans et bordures CSS](#) , B. Bos, E. Etemad, B. Kemper. W3C.

**[CSSBOX]**

[Modèle de boîte CSS](#) , E. Etemad. W3C.

**[CSSCASCADE]**

[CSS en cascade et héritage](#) , E. Etemad, T. Atkins. W3C.

**[CSSCONTAIN]**

[CSS Confinement](#) , T. Atkins, F. Rivoal, V. Levin. W3C.

**[CSSCOLOR]**

[Module couleur CSS](#) , T. Çelik, C. Lilley, L. Baron. W3C.

**[CSSCOLORADJUST]**

[Module de réglage des couleurs CSS](#) , E. Etemad, R. Atanassov, R. Lillesveen, T. Atkins. W3C.

**[CSSDEVICEADAPT]**

[CSS Device Adaptation](#) , F. Rivoal, M. Rakow. W3C.

**[AFFICHAGE CSS]**

[Affichage CSS](#) , T. Atkins, E. Etemad. W3C.

**[CSSFONTLOAD]**

[Chargement des polices CSS](#) , T. Atkins, J. Daggett. W3C.

**[FONTES CSSF]**

[Polices CSS](#) , J. Daggett. W3C.

**[CSSFLEXBOX]**

[CSS Flexible Box Layout](#) , T. Atkins, E. Etemad, R. Atanassov. W3C.

**[CSSG]**

[Contenu généré par CSS](#) , H. Lie, E. Etemad, I. Hickson. W3C.

**[CSSGRID]**

[Mise en page de la grille CSS](#) , T. Atkins, E. Etemad, R. Atanassov. W3C.

## [CSSIMAGES]

[Module Images CSS](#) , E. Etemad, T. Atkins, L. Verou. W3C.

## [CSSIMAGES4]

[CSS Images Module Niveau 4](#) , E. Etemad, T. Atkins, L. Verou. W3C.

## [LIGNE CSS]

[Mise en page CSS en ligne](#) , D. Cramer, E. Etemad. W3C.

## [LISTES CSS]

[Listes et compteurs CSS](#) , T. Atkins. W3C.

## [CSSLOGIQUE]

[Propriétés logiques CSS](#) , R. Atanassov, E. Etemad. W3C.

## [CSSMULTICOL]

[Mise en page CSS multi-colonnes](#) , H. Lie, F. Rivoal, R. Andrew. W3C.

## [CSSOM]

[Modèle d'objet de feuilles de style en cascade \(CSSOM\)](#) , S. Pieters, G. Adams. W3C.

## [VUE CSSOM]

[Module d'affichage CSSOM](#) , S. Pieters, G. Adams. W3C.

## [CSSOVERFLOW]

[Module débordement CSS](#) , L. Baron, F. Rivoal. W3C.

## [CSSPAINT]

(Non normatif) [CSS Painting API](#) , I. Kilpatrick, D. Jackson. W3C.

## [POSITIONCSS]

[Disposition positionnée CSS](#) , R. Atanassov, A. Eicholz. W3C.

## [CSSPSEUDO]

[Pseudo-éléments CSS](#) , D. Glazman, E. Etemad, A. Stearns. W3C.

## [CSSRUBY]

[Module Rubis CSS3](#) , R. Ishida. W3C.

## [CSSSCOPAGE]

[Module de portée CSS](#) , T. Atkins. W3C.

## [CSSCISING]

[Module de dimensionnement CSS intrinsèque et extrinsèque](#) , T. Atkins, E. Etemad. W3C.

## [CSSSCROLLANCHORING]

[Ancrage de défilement CSS](#) (non normatif) , T. Atkins-Bittner. W3C.

## [CSSSYNTAXE]

[Syntaxe CSS](#) , T. Atkins, S. Sapin. W3C.

## [CSSTRANSITIONS]

(Non normatif) [CSS Transitions](#) , L. Baron, D. Jackson, B. Birtles. W3C.

## [CSSTABLE]

[Tableau CSS](#) , F. Remy, G. Whitworth. W3C.

## [CSSTEXT]

[Texte CSS](#) , E. Etemad, K. Ishii. W3C.

## [VALEURS CSS]

[Valeurs et unités CSS3](#) , H. Lie, T. Atkins, E. Etemad. W3C.

## [CSSUI]

[Module d'interface utilisateur de base CSS3](#) , F. Rivoal. W3C.

## [CSWM]

[Modes d'écriture CSS](#) , E. Etemad, K. Ishii. W3C.

## [SE PRÉCIPITER]

[Streaming adaptatif dynamique sur HTTP \(DASH\)](#) . ISO.

## [POSTURE DE L'APPAREIL]

(Non normatif) [Device Posture API](#) , D. Gonzalez-Zuniga, K. Christiansen. W3C.

## [DOM]

[DOM](#) , A. van Kesteren, A. Gregor, Ms2ger. WHATWG.

## [DOMPARSING]

[Analyse et sérialisation DOM](#) , T. Leithead. W3C.

## [POINT]

(Non normatif) [Le langage DOT](#) . Graphviz.

## [E163]

*Recommandation E.163 — Plan de numérotage pour le service téléphonique international* , Livre bleu du CCITT, Fascicule II.2, pp. 128-134, novembre 1988.

## [CODAGE]

[Encodage](#) , A. van Kesteren, J. Bell. WHATWG.

## [EXECCOMMANDE]

[execCommand](#) , J. Wilm, A. Gregor. CG des API d'édition du W3C.

## [EXIF]

(Non normatif) [Format de fichier image échangeable](#) . JEITA.

## [ALLER CHERCHER]

[Fetch](#) , A. van Kesteren. WHATWG.

## [FILEAPI]

[API de fichier](#) , A. Ranganathan. W3C.

## [FILTRES]

[Effets de filtre](#) , D. Schulze, D. Jackson, C. Harrelson. W3C.

## [PLEIN ÉCRAN]

[Plein écran](#) , A. van Kesteren, T. Çelik. WHATWG.

## [GÉOMÉTRIE]

[Interfaces géométriques](#) . S. Pieters, D. Schulze, R. Cabanier. W3C.

## [GIF]

[Format d'échange graphique](#) (non normatif) . CompuServ.

## [GRAPHIQUE]

*Infographie* (non normative) : Principes et pratique en C , deuxième édition, J. Foley, A. van Dam, S. Feiner, J. Hughes. Addison-Wesley. ISBN 0-201-84840-6.

## [GRÉGORIEN]

(Non normatif) *Inter Gravissimas* , A. Lilius, C. Clavius. Bulle papale de Grégoire XIII, février 1582.

## [THS]

[Temps haute résolution](#) , I. Grigorik, J. Simonsen, J. Mann. W3C.

## [HTMLAAM]

[Mappages d'API d'accessibilité HTML 1.0](#) , S. Faulkner, A. Surkov, S. O'Hara. W3C.

## [HTTP]

[Protocole de transfert hypertexte \(HTTP/1.1\) : syntaxe et routage des messages](#) , R. Fielding, J. Reschke. IETF.

[Protocole de transfert hypertexte \(HTTP/1.1\) : sémantique et contenu](#) , R. Fielding, J. Reschke. IETF.

[Protocole de transfert hypertexte \(HTTP/1.1\) : requêtes conditionnelles](#) , R. Fielding, J. Reschke. IETF.

[Protocole de transfert hypertexte \(HTTP/1.1\) : requêtes de plage](#) , R. Fielding, Y. Lafon, J. Reschke. IETF.

[Protocole de transfert hypertexte \(HTTP/1.1\) : mise en cache](#) , R. Fielding, M. Nottingham, J. Reschke. IETF.

[Protocole de transfert hypertexte \(HTTP/1.1\) : Authentification](#) , R. Fielding, J. Reschke. IETF.

## [INDEXEDDB]

[API de base de données indexée](#) , A. Alabbas, J. Bell. W3C.

## [INBAND]

[Sourcer des pistes de ressources multimédias intrabande à partir de conteneurs multimédias vers HTML](#) , S. Pfeiffer, B. Lund. W3C.

## [INFRA]

[Infra](#) , A. van Kesteren, D. Denicola. WHATWG.

**[OBSERVATEUR D'INTERSECTION]**

[Observateur d'intersection](#) , S. Zager. W3C.

**[RESIZEOBSERVER]**

[Redimensionner l'Observateur](#) , O. Brufau, E. Álvarez. W3C.

**[ISO3166]**

[ISO 3166 : Codes pour la représentation des noms de pays et de leurs subdivisions](#) . ISO.

**[ISO4217]**

[ISO 4217 : Codes pour la représentation des monnaies et des fonds](#) . ISO.

**[ISO8601]**

(Non normatif) [ISO8601 : Éléments de données et formats d'échange — Échange d'informations — Représentation des dates et des heures](#) . ISO.

**[JAVASCRIPT]**

[Spécification du langage ECMAScript](#) . Ecma International.

**[JLREQ]**

[Exigences pour la mise en page du texte japonais](#) . W3C.

**[JPEG]**

[Format d'échange de fichiers JPEG](#) , E. Hamilton.

**[JSERRORSTACKS]**

[Piles d'erreurs](#) (non normatives) . Ecma International.

**[JSIMPORTASSERTIONS]**

[Importer des assertions](#) . Ecma International.

**[JSJSONMODULES]**

[Modules JSON](#) . Ecma International.

**[JSRESIZABLEBUFFERS]**

[ArrayBuffer redimensionnable et SharedArrayBuffer évolutif](#) . Ecma International.

**[JSINTL]**

[Spécification de l'API d'internationalisation ECMAScript](#) . Ecma International.

**[JSON]**

[Le format d'échange de données JSON \(JavaScript Object Notation\)](#) , T. Bray. IETF.

**[TÂCHES LONGUES]**

[Tâches longues](#) , D. Denicola, I. Grigorik, S. Panicker. W3C.

**[MAILTO]**

(Non normatif) [Le schéma d'URI 'mailto'](#) , M. Duerst, L. Masinter, J. Zawinski. IETF.

#### **[MANIFESTE]**

[Manifeste d'application Web](#) , M. Caceres, K. Rohde Christiansen, M. Lamouri, A. Kostianen, M. Giuca, A. Gustafson. W3C.

#### **[MATHML]**

[Langage de balisage mathématique \(MathML\)](#) , D. Carlisle, P. Ion, R. Miner. W3C.

#### **[MÉDIAFRAG]**

[Fragments de média URI](#) , R. Troncy, E. Mannens, S. Pfeiffer, D. Van Deursen. W3C.

#### **[MEDIASOURCE]**

[Extensions de source multimédia](#) , A. Colwell, A. Bateman, M. Watson. W3C.

#### **[FLUX MÉDIA]**

[Capture multimédia et flux](#) , D. Burnett, A. Bergkvist, C. Jennings, A. Narayanan. W3C.

#### **[SIGNALEMENT]**

[Reportage](#) , D. Creager, I. Clelland, M. West. W3C.

#### **[MFREL]**

[Wiki Microformats : valeurs rel existantes](#) . Microformats.

#### **[MIMESNIFF]**

[MIME Sniffing](#) , G. Hemsley. WHATWG.

#### **[MÉLANGER]**

[Contenu mixte](#) , M. West. W3C.

#### **[MNG]**

[Format MNG \(Multiple-image Network Graphics\)](#) . G. Randers-Pehrson.

#### **[MPEG2]**

*ISO/CEI 13818-1 : Technologies de l'information — Codage générique des images animées et des informations audio associées : Systèmes* . ISO/CEI.

#### **[MPEG4]**

*ISO/IEC 14496-12 : format de fichier multimédia de base ISO* . ISO/CEI.

#### **[QM]**

[Media Queries](#) , H. Lie, T. Çelik, D. Glazman, A. van Kesteren. W3C.

#### **[TAMPON MULTIPLE]**

(Non normatif) [Mise en mémoire tampon multiple](#) . Wikipédia.

#### **[TEMPS DE NAVIGATION]**

[Chronométrage de la navigation](#) , Y. Weiss. W3C.

## [NPAPI]

(Non normatif) [Référence de l'API du plugin Gecko](#) . Mozilla.

## [OGGSKELETONHEADERS]

[En-têtes de squelette](#) . Xiph.Org.

## [OUVRIR RECHERCHE]

[Découverte automatique en HTML/XHTML](#) . Dans *OpenSearch 1.1 Brouillon* 6 . GitHub.

## [ORIGINE]

(Non normatif) [The Web Origin Concept](#) , A. Barth. IETF.

## [PEINTURE]

[Calendrier de peinture](#) , S. Panicker. W3C.

## [DEMANDE DE PAIEMENT]

[API de demande de paiement](#) , M. Cáceres, D. Wang, R. Solomakhin, I. Jacobs. W3C.

## [PDF]

(Non normatif) [Gestion de documents — Format de document portable — Partie 1 : PDF](#) . ISO.

## [POLITIQUE D'AUTORISATION]

[Politique d'autorisations](#) , I. Clelland, W3C.

## [PINGBACK]

[Pingback 1.0](#) , S. Langridge, I. Hickson.

## [PNG]

[Spécification des graphiques réseau portables \(PNG\)](#) , D. Duce. W3C.

## [POINTERÉVÉNEMENTS]

[Événements de pointeur](#) , J. Rossi, M. Brubeck, R. Byers, PH Lauke. W3C.

## [POINTERLOCK]

[Pointer Lock](#) , V. Scheib. W3C.

## [PPUTF8]

(Non normatif) [Les propriétés et les promesses d'UTF-8](#) , M. Dürst. Université de Zurich. Dans *Actes de la 11e Conférence internationale Unicode* .

## [PRÉCHARGER]

[Précharge](#) , I. Grigorik. W3C.

## [PRÉSENTATION]

[Présentation API](#) , M. Foltz, D. Röttches. W3C.

## [POLITIQUE DE RÉFÉRENCE]

[Politique de parrainage](#) , J. Eisinger, E. Stark. W3C.

## **[DEMANDE DE RAPPEL]**

[Ordonnancement coopératif des tâches d'arrière-plan](#) , R. McIlroy, I. Grigorik. W3C.

## **[CONSEILS DE RESSOURCES]**

[Conseils de ressources](#) , I. Grigorik. W3C.

## **[RFC1034]**

[Noms de domaine - Concepts et équipements](#) , P. Mockapetris. IETF, novembre 1987.

## **[RFC1123]**

[Exigences pour les hôtes Internet -- Application et assistance](#) , R. Braden. IETF, octobre 1989.

## **[RFC2046]**

[Extensions de messagerie Internet polyvalentes \(MIME\) Deuxième partie : Types de média](#) , N. Freed, N. Borenstein. IETF.

## **[RFC2397]**

[Le schéma d'URL "data"](#) , L. Masinter. IETF.

## **[RFC5545]**

[Spécification de l'objet de base de l'agenda et de la planification Internet \(iCalendar\)](#) , B. Desruisseaux. IETF.

## **[RFC2483]**

[Services de résolution d'URI nécessaires à la résolution d'URN](#) , M. Mealling, R. Daniel. IETF.

## **[RFC3676]**

[Le format Text/Plain et les paramètres DelSp](#) , R. Gellens. IETF.

## **[RFC9239]**

[Mises à jour des types de médias ECMAScript](#) , M. Miller, M. Borins, M. Bynens, B. Farias. IETF.

## **[RFC4337]**

(Non normatif) [Enregistrement de type MIME pour MPEG-4](#) , Y. Lim, D. Singer. IETF.

## **[RFC7595]**

[Lignes directrices et procédures d'enregistrement pour les schémas d'URI](#) , D. Thaler, T. Hansen, T. Hardie. IETF.

## **[RFC5322]**

[Format de message Internet](#) , P. Resnick. IETF.

## **[RFC6381]**

[Les paramètres 'Codecs' et 'Profiles' pour les types de médias "Bucket"](#) , R. Gellens, D. Singer, P. Frojdh. IETF.



**[RFC6266]**

[\*Utilisation du champ d'en-tête Content-Disposition dans le protocole de transfert hypertexte \(HTTP\)\*](#) , J. Reschke. IETF.

**[RFC6350]**

[\*Spécification du format vCard\*](#) , S. Perreault. IETF.

**[RFC6596]**

[\*La relation de lien canonique\*](#) , M. Ohye, J. Kupke. IETF.

**[RFC7034]**

(Non normatif) [\*Champ d'en-tête HTTP X-Frame-Options\*](#) , D. Ross, T. Gondrom. IETF.

**[RFC7303]**

[\*Types de média XML\*](#) , H. Thompson, C. Lilley. IETF.

**[RFC7578]**

[\*Renvoyer des valeurs à partir de formulaires : multipart/form-data\*](#) , L. Masinter. IETF.

**[RFC8297]**

[\*Un code d'état HTTP pour indiquer des indices\*](#) , K. Oku. IETF.

**[ORIENTATION DE L'ÉCRAN]**

[\*Orientation de l'écran\*](#) , M. Cáceres. W3C.

**[SCSU]**

(Non normatif) [\*UTR #6 : Un schéma de compression standard pour Unicode\*](#) , M. Wolf, K. Whistler, C. Wicksteed, M. Davis, A. Freytag, M. Scherer. Consortium Unicode.

**[CONTEXTES SÉCURISÉS]**

[\*Contextes sécurisés\*](#) , M. West. W3C.

**[SÉLECTION]**

[\*API de sélection\*](#) , R. Niwa. W3C.

**[SÉLECTEURS]**

[\*Sélecteurs\*](#) , E. Etemad, T. Çelik, D. Glazman, I. Hickson, P. Linss, J. Williams. W3C.

**[SMS]**

(Non normatif) [\*URI Scheme for Global System for Mobile Communications \(GSM\) Short Message Service \(SMS\)\*](#) , E. Wilde, A. Vaha-Sipila. IETF.

**[CHAMPS STRUCTURÉS]**

[\*Valeurs de champ structuré pour HTTP\*](#) , M. Nottingham, PH. Camp. IETF.

**[ISR]**

[\*Intégrité des sous-ressources\*](#) , D. Akhawe, F. Braun, F. Marier, J. Weinberger. W3C.

## [STOCKAGE]

[Stockage](#) , A. van Kesteren. WHATWG.

## [SVG]

[Graphiques vectoriels évolutifs \(SVG\) 2](#) , N Andronikos, R. Atanassov, T. Bah, B. Birtles, B. Brinza, C. Concolato, E. Dahlström, C. Lilley, C. McCormack, D. Schepers, R. Schwerdtfeger , D. Storey, S. Takagi, J. Watt. W3C.

## [SW]

[Travailleurs des services](#) , A. Russell, J. Song, J. Archibald. W3C.

## [TOR]

(Non normatif) [Tor](#) .

## [TOUCHER]

[Événements tactiles](#) , D. Schepers, S. Moon, M. Brubeck, A. Barstow, R. Byers. W3C.

## [ZDATABASE]

(Non normatif) [Base de données des fuseaux horaires](#) . IANA.

## [UAAG]

(Non normatif) [User Agent Accessibility Guidelines \(UAAG\) 2.0](#) , J. Allan, K. Ford, J. Richards, J. Spellman. W3C.

## [UIÉVÉNEMENTS]

[Spécification des événements d'interface utilisateur](#) , G. Kacmarcik, T. Leithead. W3C.

## [UNICODE]

[La norme Unicode](#) . Consortium Unicode.

## [UNIVCHARDET]

(Non normatif) [Une approche composite de la détection langage/encodage](#) , S. Li, K. Momoi. Netscape. Dans *Actes de la 19e Conférence internationale Unicode* .

## [URL]

[URL](#) , A. van Kesteren. WHATWG.

## [URNE]

[Syntaxe URN](#) , R. Moats. IETF.

## [UTF7]

(Non normatif) [UTF-7 : Un format de transformation Mail-Safe d'Unicode](#) , D. Goldsmith, M. Davis. IETF.

## [UTF8DET]

[Codage de forme multilingue](#) (non normatif) , M. Dürst. W3C.

## [UTR36]

(Non normatif) [\*UTR #36 : Unicode Security Considerations\*](#) , M. Davis, M. Suignard. Consortium Unicode.

#### [WASMJS]

(Non normatif) [\*WebAssembly JavaScript Interface\*](#) , D. Ehrenberg. W3C.

#### [WCAG]

(Non-normative) [\*Web Content Accessibility Guidelines \(WCAG\)\*](#) , A. Kirkpatrick, J. O Connor, A. Campbell, M. Cooper. W3C.

#### [WEBANIMATIONS]

[\*Animations Web\*](#) , B. Birtles, S. Stephens, D. Stockwell. W3C.

#### [WEBAUDIO]

(Non normatif) [\*Web Audio API\*](#) , P. Adenot, H. Choi. W3C.

#### [WEBAUTHN]

[\*Authentification Web : une API pour accéder aux identifiants de clé publique\*](#) , M. Jones, A. Kumar, E. Lundberg, D. Balfanz, V. Bharadwaj, A. Birgisson, A. Czeskis, J. Hodges, JC Jones, H. Le Van Gong, A. Liao, R. Lindemann, J. Bradley, C. Brand, T. Cappalli, A. Langley, G. Mandyam, M. Miller, N. Satragno, N. Steele, J. Tan, S. Weeden, M. West, J. Yasskin. W3C.

#### [WEBCODECS]

[\*API WebCodecs\*](#) , C. Cunningham, P. Adenot, B. Aboba. W3C.

#### [WEBCRYPTO]

[\*API de cryptographie Web\*](#) , D. Huigens. W3C.

#### [PILOTE WEB]

[\*Pilote Web\*](#) . W3C

#### [PILOTE WEBBIDI]

[\*Pilote Web BiDi\*](#) . W3C

#### [WEBGL]

[\*Spécifications WebGL\*](#) , D. Jackson, J. Gilbert. Groupe Khronos.

#### [WEBGPU]

[\*WebGPU\*](#) , D. Malyshau, K. Ninomiya. W3C.

#### [WEBIDL]

[\*Web IDL\*](#) , E. Chen, T. Gu. WHATWG.

#### [LIEN WEB]

[\*Liens Web\*](#) , M. Nottingham. IETF.

#### [WEBMCG]

[\*Directives sur les conteneurs WebM\*](#) . Le projet WebM.

#### [WEBNFC]

(Non normatif) [\*Web NFC\*](#) , F. Beaufort, K. Christiansen, Z. Kis. W3C.

## [WEBSOCKETS]

[WebSockets](#) , A. Rice. WHATWG.

## [WEBVTT]

[WebVTT](#) , S. Pieters. W3C.

## [WHATWGWIKI]

[Le wiki WHATWG](#) . WHATWG.

## [X121]

*Recommandation X.121 — Plan de numérotage international pour les réseaux publics pour données* , Livre bleu du CCITT, Fascicule VIII.3, pp. 317-332.

## [XFN]

[Profil XFN 1.1](#) , T. Çelik, M. Mullenweg, E. Meyer. GMPG.

## [XHR]

[XMLHttpRequest](#) , A. van Kesteren. WHATWG.

## [XKCD1288]

[Substitutions](#) (non normatives) , Randall Munroe. xkcd.

## [XML]

[Langage de balisage extensible](#) , T. Bray, J. Paoli, C. Sperberg-McQueen, E. Maler, F. Yergeau. W3C.

## [XMLENTITÉ]

[Définitions d'entités XML](#) (non normatives) pour les caractères , D. Carlisle, P. Ion. W3C.

## [XMLNS]

[Espaces de noms en XML](#) , T. Bray, D. Hollander, A. Layman, R. Tobin. W3C.

## [XMLSSPI]

[Association des feuilles de style aux documents XML](#) , J. Clark, S. Pieters, H. Thompson. W3C.

## [XPATH10]

[Langage de chemin XML \(XPath\) version 1.0](#) , J. Clark, S. DeRose. W3C.

## [XSLT10]

Transformations XSL (non normatives) [\(XSLT\) Version 1.0](#) , J. Clark. W3C.

## [XSLTP]

(Non normatif) [DOM XSLTProcessor](#) , WHATWG Wiki. WHATWG.

1. [Remerciements](#)
2. [Droits de propriété intellectuelle](#)

## Remerciements

Merci à Tim Berners-Lee d'avoir inventé le HTML, sans lequel rien de tout cela n'existerait.

Merci à Aankhen, Aaq Ishtyaq, Aaron Boodman, Aaron Leventhal, Aaron Krajeski, Abhishek Ghaskata, Abhishek Gupta, Adam Barth, Adam de Boor, Adam Hepton, Adam Klein, Adam Rice, Adam Roben, Addison Phillips, Adele Peterson, Adrian Bateman, Adrian Roselli, Adrian Sutton, Agustín Fernández, Aharon (Vladimir) Lanin, Ajai Tirumali, Ajay Poshak, Akatsuki Kitamura, Alan Plum, Alastair Campbell, Alejandro G. Castro, Alex Bishop, Alex Nicolaou, Alex Nozdriukhin, Alex Rousskov, Alex Soncodi, Alexander Farkas, Alexander J. Vincent, Alexandre Dieulot, Alexandre Morgaut, Alexey Feldgendler, Алексей Проскуряков (Alexey Proskuryakov), Alexey Shvayka, Alexis Deveria, Alfred Agrell, Ali Juma, Alice Boxhall, Alice Wonder, Allan Clements, Allen Wirfs-Brock, Alex Komoroske, Alex Russell, Alphan Chen, Aman Ansari, Ami Fischman, Amos Jeffries, Amos Lim, Anders Carlsson, André Bargull, André E. Veltstra, Andrea Rendine, Andreas, Andreas Deutschlinger, Andreas Kling, Andrei Popescu, Andres Gomez, Andres Rios, Andreu Botella, Andrew Barfield, Andrew Clover, Andrew Gove, Andrew Grieve, Andrew Macpherson, Andrew Oakley, Andrew Paseltiner, Andrew Simons, Andrew Smith, Andrew W. Hagen, Andrew Williams, Andrey V. Lukyanov, Andry Rendy, Andy Davies , Andy Earnshaw, Andy Heydon, Andy Paicu, Andy Palay, Anjana Vakil, Ankur Kaushal, Anna Belle Leiserson, Anna Sidwell, Anthony Boyd, Anthony Bryan, Anthony Hickson, Anthony Ramine, Anthony Ricaud, Anton Vayvod, Antonio Sartori, Antti Koivisto, Arfat Salman, Arkadiusz Michalski, Arne Thomassen, Aron Spohr, Arphen Lin, Arthur Hemery, Arthur Sonzogni, Arthur Stolyar, Arun Patole, Aryeh Gregor, Asanka Herath, Asbjørn Ulsberg, Ashley Gullen, Ashley Sheridan, Asumu Takikawa, Atsushi Takayama, Attila Haraszti, Aurélien Levy , Ave Wrigley, Avi Drissman, Axel Dahmen, 방성범 (Bang Seongbeom), Ben Boyle, Ben Godfrey, Ben Golightly, Ben Kelly, Ben Lerner, Ben Leslie, Ben Meadowcroft, Ben Millard, Benjamin Carl Wiley Sittler, Benjamin Hawkes-Lewis , Benji Bilheimer, Benoit Ren, Bert Bos, Bijan Parsia, Bill Corry, Bill Mason, Bill McCoy, Billy Wong, Billy Woods, Bjartur Thorlacius, Björn Höhrmann, Blake Frantz, Bob Lund, Bob Owen, Bobby Holley, Boris Zbarsky, Brad Fults, Brad Neuberg, Brad Spencer, Bradley Meck, Brady Eidson, Brandon Jones, Brendan Eich, Brenton Simpson, Brett Wilson, Brett Zamir, Brian Birtles, Brian Blakely, Brian Campbell, Brian Korver, Brian Kuhn, Brian M. Dube, Brian Ryner, Brian Smith, Brian Wilson, Bryan Sullivan, Bruce Bailey, Bruce D'Arcus, Bruce Lawson, Bruce Miller, Bugs Nash, C. Scott Ananian, C. Williams, Cameron McCormack, Cameron Zemek, Cao Yipeng, Carlos Amengual, Carlos Gabriel Cardona, Carlos Ibarra López, Carlos Perelló Marín, Carolyn MacLeod, Casey Leask, Cătălin Badea, Cătălin Mariș, Cem Turesoy, caving, Chao Cai, 윤석찬 (Channy Yun), Charl van Niekerk, Charlene Wright, Charles Iliya Krempeaux, Charles McCathie Nevile, Charlie Reis, 白丞祐 (Cheng-You Bai ), Chris Apers, Chris Cressman, Chris Dumez, Chris Evans, Chris Harrelson, Chris Markiewicz, Chris Morris, Chris Nardi, Chris Needham, Chris Pearce, Chris Peterson, Chris Rebert, Chris Weber, Chris Wilson, Christian Biesinger, Christian Johansen, Christian Schmidt, Christoph Päper, Christophe Dumez, Christopher Aillon, Christopher Cameron , Christopher Ferris, Chriswa, Clark Buehler, Cole Robison, Colin Fine, Collin Jackson, Corey Farwell, Corprew Reed, Craig Cockburn, Csaba Gabor, Csaba

Marton, Cynthia Shelly, Cyrille Tuzi, Daksh Shah, Dan Callahan, Dan Yoder, Dane Foster , Daniel Barclay, Daniel Bratell, Daniel Brooks, Daniel Brumbaugh Keeney, Daniel Buchner, Daniel Cheng, Daniel Clark, Daniel Davis, Daniel Ehrenberg, Daniel Glazman, Daniel Holbert, Daniel Peng, Daniel Schattenkirchner, Daniel Spång, Daniel Steinberg, Daniel Tan, Daniel Trebbien, Daniel Vogelheim, Danny Sullivan, Daphne Preston-Kendal, Darien Maillet Valentine, Darin Adler, Darin Fisher, Darxus, Dave Camp, Dave Cramer, Dave Hodder, Dave Lampton, Dave Singer, Dave Tapuska, Dave Townsend, David Baron, David Bloom, David Bruant, David Carlisle, David E. Cleary, David Egan Evans, David Fink, David Flanagan, David Gerard, David Grogan, David Hale, David Håsäther, David Hyatt, David I. Lehn, David John Burrowes, David Matja, David Remahl, David Resseguie, David Smith, David Storey, David Vest, David Woolley, David Zbarsky, Dave Methvin, DeWitt Clinton, Dean Edridge, Dean Edwards, Dean Jackson, Debanjana Sarkar, Debi Orton, Delan Azabani, Derek Featherstone, Derek Guenther, Devarshi Pant, Devdatta, Devin Rouso, Diego Ferreira Val, Diego González Zúñiga, Diego Ponce de León, Dimitri Glazkov, Dmitry Golubovsky, Dirk Pranke, Dirk Schulze, Dirkjan Ochtman, Divya Manian, Dmitry Lazutkin, Dmitry Titov, dauphin, Dominic Cooney, Dominic Farolino, Dominique Hazaël-Massieux, Don Brutzman, Donovan Glover, Doron Rosenberg, Doug Kramer, Doug Simpkinson, Drew Wilson, Edgar Chen, Edmund Lai, Eduard Pascual, Eduardo Vela, Edward Welbourne, Edward Z. Yang, Ehsan Akhgari, Eira Monstad, Eitan Adler, Eli Friedman, Eli Gray, Eliot Graff, Elisabeth Robson, Elizabeth Castro, Elliott Sprehn, Elliotte Harold, Emilio Cobos Álvarez, Emily Stark, Eric Carlson, Eric Casler, Eric Lawrence, Eric Portis, Eric Rescorla, Eric Semling ,Eric Willigers, Erik Arvidsson, Erik Charlebois, Erik Rose, 栗本 英理子 (Eriko Kurimoto), espreto, Evan Jacobs, Evan Martin, Evan Prodromou, Evan Stade, Evert, Evgeny Kapun, ExE-Boss, Ezequiel Garzón, fantasai, Félix Sanz, Felix Sasaki, Fernando Altomare Serboncini, Forbes Lindesay, Francesco Schwarz, Francis Brosnan Blazquez, Franck 'Shift' Quélain, François Marier, Frank Barchard, Frank Liberato, Franklin Shirley, Fredrik Söderquist, 鵜飼文敏 (Fumitoshi Ukai), Futomi Hatano, Gavin Carothers , Gavin Kistner, Gareth Rees, Garrett Smith, Gary Blackwood, Gary Kacmarcik, Gary Katsevman, Geoff Richards, Geoffrey Garen, Georg Neis, George Lund, Gianmarco Armellin, Giovanni Campagna, Giuseppe Pascale, Glenn Adams, Glenn Maynard, Graham Klyne, Greg Botten, Greg Houston, Greg Wilkins, Gregg Tavares, Gregory J. Rosmaita, Gregory Terzian, Grey, Guilherme Johansson Tramontina, guest271314, Gytis Jakutonis, Håkon Wium Lie, Habib Virji, Hajime Morrita, Hallvord Reiar Michaelsen Steen, Hanna Laakso, Hans S. Tømmerhalt , Hans Stimer, Harald Alvestrand, Hayato Ito, 何志翔 (HE Zhixiang), Henri Sivonen, Henrik Lied, Henry Lewis, Henry Mason, Henry Story, Hermann Donfack Zeufack, 中川博貴 (Hiroki Nakagawa), Hiroshige Hayashizaki, Hiroyuki USHITO, Hitoshi Yoshida , Hongchan Choi, 王华 (Hua Wang), Hugh Bellamy, Hugh Guiney, Hugh Winkler, Ian Bicking, Ian Clelland, Ian Davis, Ian Fette, Ian Henderson, Ian Kilpatrick, Ibrahim Ahmed, Ido Green, Ignacio Javier, Igor Oliveira, 安次嶺 一功 (Ikko Ashimine), Ingvar Stepanyan, isonmad, Iurii Kucherov, Ivan Enderlin, Ivan Nikulin, Ivan Panchenko, Ivo Emanuel Gonçalves, J. King, JC Jones, Jackson Ray Hamilton, Jacob Davies, Jacques Distler, Jake Archibald, Jake Verbaten, Jakub Vrána, Jakub Łopuszański, Jakub Wilk, James Craig , James Graham, James Greene, James Justin Harrell, James Kozianski, James M Snell, James Perrett, James Robinson, Jamie Liu, Jamie Lokier, Jan Kühle, Jan Miksovsky, Janice Shiu, Janusz Majnert, Jan-Ivar Bruaroey, Jan-Klaas Kollhof, Jared Jacobs, Jason Duell, Jason Kersey, Jason Lustig, Jason

Orendorff, Jason White, Jasper Bryant-Greene, Jasper St. Pierre, Jatinder Mann, Jean-Yves Avenard, Jed Hartman, Jeff Balogh, Jeff Cutsinger, Jeff Gilbert, Jeff "JeffH" Hodges, Jeff Schiller, Jeff Walden, Jeffrey Yasskin, Jeffrey Zeldman, 胡慧鋒 (Jennifer Braithwaite), Jellybean Stonerfish, Jennifer Apacible, Jens Bannmann, Jens Fendler, Jens Oliver Meiert, Jens Widell, Jer Noble, Jeremy Hustman, Jeremy Keith, Jeremy Orlow, Jeremy Roman, Jeroen van der Meer, Jerry Smith, Jesse Renée Beach, Jessica Jong, jfkthame, Jian Li, Jihye Hong, Jim Jewett, Jim Ley, Jim Meehan, Jim Michaels, Jinho Bang, Jinjiang (勾三股四), Jirka Kosek, Jjgod Jiang, Joaquim Medeiros, João Eiras, Jochen Eisinger, Joe Clark, Joe Gregorio, Joel Spolsky, Joel Verhagen, Joey Arhar, Johan Herland, Johanna Herman, John Boyer, John Bussjaeger, John Carpenter, John Daggett, John Fallows, John Foliot, John Harding, John Keizer, John Law, John Musgrave, John Snyders, John Stockton, John-Mark Bell, Johnny Stenback, Jon Coppeard, Jon Ferraiolo, Jon Gibbins, Jon Jensen, Jon Perlow, Jonas Sicking, Jonathan Cook, Jonathan Neal, Jonathan Oddy, Jonathan Rees, Jonathan Watt, Jonathan Worent, Jonny Axelsson, Joram Schrijver, Jordan Tucker, Jorgen Horstink, Joris van der Wel, Jorunn Danielsen Newth, Joseph Kesselman, Joseph Mansfield, Joseph Pecoraro, Josh Aas, Josh Hart, Josh Juran, Josh Levenberg, Josh Matthews, Joshua Bell, Joshua Randall, Juan Olvera, Juanmi Huertas, Jukka K. Korpela, Jules Clément-Ripoche, Julian Reschke, Julio Lopez, 小勝 純 (Jun Kokatsu), Jun Yang (harttle), Jungkee Song, Jürgen Jeka, Justin Lebar, Justin Novosad, Justin Rogers, Justin Schuh, Justin Sinclair, Juuso Lapinlampi, Ka-Sing Chou, Kagami Sascha Rosylight, Kai Hendry, Kamishetty Sreeja, 呂康豪 (KangHao Lu), Karl Dubost, Karl Tomlinson, Kartik Arora, Kartikaya Gupta, 葛依寧 (Kat Hackett), Kathy Walton, 河童エクマ (Kawarabe Ecma) Keith Rollin, Keith Yeung, Kelly Ford, Kelly Norton, Ken Russell, Kenji Baheux, Kevin Benson, Kevin Cole, Kevin Gadd, Kevin Venkiteswaran, Kinuko Yasuda, Koji Ishii, Kornél Pál, Kornel Lesinski, 上野 康平 (UENO, Kouhei), Kris Northfield, Kristof Zelechovski, Krzysztof Maczyński, 黒澤剛志 (Kurosawa Takeshi), Kyle Barnhart, Kyle Hofmann, Kyle Huey, Léonard Bouchet, Léonie Watson, Lachlan Hunt, Larry Masinter, Larry Page, Lars Gunther, Lars Solberg, Laura Carlson, Laura Granka, Laura L. Carlson, Laura Wisewell, Laurens Holst, Lawrence Forooghian, Lee Kowalkowski, Leif Halvard Silli, Leif Kornstaedt, Lenny Domnitser, Leonard Rosenthol, Leons Petrazickis, Liviu Tinta, Lobotom Dysmon, Logan, Logan Moore, Loune, Lucas Gadani, Łukasz Pilorz, Luke Kenneth Casson Leighton, Maciej Stachowiak, Magne Andersson, Magnus Kristiansen, Maik Merten, Majid Valipour, Malcolm Rowe, Manish Goregaokar, Manish Tripathi, Manuel Martinez-Almeida, Manuel Rego Casasnovas, Marc Hoyois, Marc-André Choquette, Marc-André Lafortune, Marco Zehe, Marcus Bointon, Marijn Kruisselbrink, Mark Amery, Mark Birbeck, Mark Davis, Mark Green, Mark Miller, Mark Nottingham, Mark Pilgrim, Mark Rogers, Mark Rowe, Mark Schenk, Mark Vickers, Mark Wilton-Jones, Markus Cadonau, Markus Stange, Martijn van der Ven, Martijn Wargers, Martin Atkins, Martin Chaov, Martin Dürst, Martin Honnen, Martin Janecke, Martin Kutschker, Martin Nilsson, Martin Thomson, Masataka Yakura, Masatoshi Kimura, Mason Freed, Mason Mize, Mathias Bynens, Mathieu Henri, Matias Larsson, Matt Brubeck, Matt Di Pasquale, Matt Falkenhagen, Matt Schmidt, Matt Wright, Matthew Gaudet, Matthew Grogan, Matthew Mastracci, Matthew Noorenberghe, Matthew Raymond, Matthew Thomas, Matthew Tylee Atkinson, Mattias Waldau, Max Romantschuk, Maxim Tsoy, Mayeul Cantan, Menachem Salomon, Menno van Slooten, Micah Dubinko, Micah Nerren, Michael 'Ratt' Iannarelli, Michael A. Nachbaur, Michael A. Puls II, Michael

Carter, Michael Daskalov, Michael Day, Michael Dyck, Michael Enright, Michael Gratton, Michael Kohler, Michael McKelvey, Michael Nordman, Michael Powers, Michael Rakowski, Michael(tm) Smith, Michael Walmsley, Michal Zalewski, Michel Buffa, Michel Fortin, Michelangelo De Simone, Michiel van der Blonk, Miguel Casas-Sanchez, Mihai Şucan, Mihai Parparita, Mike Brown, Mike Dierken, Mike Dixon, Mike Hearn, Mike Pennisi, Mike Schinkel, Mike Shaver, Mikko Rantalainen, Mingye Wang, Mohamed Zergaoui, Mohammad Al Houssami, Mohammad Reza Zakerinasab, Momdo Nakamura, Morten Stenshorne, Mounir Lamouri, Ms2ger, mtrootyy, 邱慕安 (Mu-An Chiou), Mukilan Thiyagarajan, Mustaq Ahmed, Myles Borins, Nadia Heninger, NARUSE Yui, Navid Zolghadr, Neil Deakin, Neil Rashbrook, Neil Soiffer, Nereida Rondon, networkException, Nicholas Shanks, Nicholas Stimpson, Nicholas Zakas, Nickolay Ponomarev, Nicolas Gallagher, Nicolas Pena Moreno, Nicolò Ribaud, Nikki Bee, Niklas Gögge, Nina Satragno, Noah Mendelsohn, Noah Slater, Noam Rosenthal, Noel Gordon, Nolan Waite, NoozNooz42, Norbert Lindenberg, Oisín Nolan, Ojan Vafai, Olaf Hoffmann, Olav Junker Kjær, Oldřich Vetešník, Oli Studholme, Oliver Hunt, Oliver Rigby, Olivia (Xiaoni) Lai, Olivier Gendrin, Olli Pettay, Ondřej Žára, Ori Avtalion, Oriol Brufau, oSand, Pablo Flouret, Patrick Dark, Patrick Garies, Patrick H. Lauke, Patrik Persson, Paul Adenot, Paul Lewis, Paul Norman, Per-Erik Brodin, —丝 (percyley), Perry Smith, Peter Beverloo, Peter Karlsson, Peter Kasting, Peter Moulder, Peter Occil, Peter Stark, Peter Van der Beken, Peter van der Zee, Peter-Paul Koch, Phil Pickering, Philip Ahlberg, Philip Brembeck, Philip Taylor, Philip TAYLOR, Philippe De Ryck, Pierre-Arnaud Allumé, Pierre-Marie Dartus, Piers Wombwell, Pooja Sanklecha, Prashant Hiremath, Prashanth Chandra, Prateek Rungta, Pravir Gupta, Prayag Verma, 李普君 (Pujun Li), Rachid Finge, Rafael Weinstein, Rafał Miłecki, Rahul Purohit, Raj Doshi, Rajas Moonka, Rakina Zata Amni, Ralf Stoltze, Ralph Giles, Raphael Champeimont, Rebecca Star, Remci Mizkur, Remco, Remy Sharp, René Saarsoo, René Stach, Ric Hardacre, Rich Clark, Rich Doughty, Richa Rupela, Richard Gibson, Richard Ishida, Ricky Mondello, Rigo Wenning, Rikkert Koppes, Rimantas Liubertas, Riona Macnamara, Rob Buis, Rob Ennals, Rob Jellinghaus, Rob S, Rob Smith, Robert Blaut, Robert Collins, Robert Hogan, Robert Kieffer, Robert Linder, Robert Millan, Robert O'Callahan, Robert Sayre, Robin Berjon, Robin Schaufler, Rodger Combs, Roland Steiner, Roma Matusевич, Romain Deltour, Roman Ivanov, Roy Fielding, Rune Lillesveen, Russell Bicknell, Ruud Steltenpool, Ryan King, Ryan Landay, Ryan Sleevi, Ryo Kajiura, Ryo Kato, Ryosuke Niwa, S. Mike Dierken, Salvatore Loreto, Sam Dutton, Sam Kuper, Sam Ruby, Sam Sneddon, Sam Weinig, Samikshya Chand, Samuel Bronson, Samy Kamkar, Sander van Lambalgen, Sanjoy Pal, Sarah Gebauer, Sarven Capadisli, Satrujit Behera, Schalk Neethling, Scott Beardsley, Scott González, Scott Hess, Scott Miles, Scott O'Hara, Sean B. Palmer, Sean Feng, Sean Fraser, Sean Hayes, Sean Hogan, Sean Knapp, Sebastian Markbåge, Sebastian Schnitzenbaumer, Sendil Kumar N, Seth Call, Seth Dillingham, Shannon Moeller, Shanti Rao, Shaun Inman, Shiino Yuki, 贺师俊 (HE Shi-Jun), Shiki Okasaka, Shivani Sharma, shreyateeza, Shubheksha Jalan, Sidak Singh Aulakh, Sierk Bornemann, Sigbjørn Finne, Sigbjørn Vik, Silver Ghost, Silvia Pfeiffer, Šime Vidas, Simon Fraser, Simon Montagu, Simon Sapin, Yu Han, Simon Spiegel, skeww, Smylers, Srirama Chandra Sekhar Mogali, Stanton McCandlish, stasoid, Stefan Håkansson, Stefan Haustein, Stefan Santesson, Stefan Schumacher, Ştefan Vargyas, Stefan Weiss, Steffen Meschkat, Stephen Ma, Stephen Stewart, Stephen White, Steve Comstock, Steve Faulkner, Steve Orvell, Steve Runyon, Steven Bennett, Steven Bingler, Steven



Garrity, Steven Tate, Stewart Brodie, Stuart Ballard, Stuart Langridge, Stuart Parmenter, Subramanian Peruvemba, Sudhanshu Jaiswal, sudokus999, Sunava Dutta, Surma, Susan Borgrink, Susan Lesch, Sylvain Pasche, TJ Crowder, Tab Atkins-Bittner, Taiju Tsuiki, Takashi Toyoshima, Takayoshi Kochi, Takeshi Yoshino, Tanteke Çelik, 田村健人 (Kent TAMURA), Taylor Hunt, Ted Mielczarek, Terence Eden, Terrence Wood, Tetsuharu OHZEKI, Theresa O'Connor, Thijs van der Vossen, Thomas Broyer, Thomas Koetter, Thomas O'Connor, Tim Altman, Tim Dresser, Tim Johansson, Tim Nguyen, Tim Perry, Tim van der Lippe, TJ VanToll, Tobias Schneider, Tobie Langel, Toby Inkster, Todd Moody, Tom Baker, Tom Pike, Tom Schuster, Tom ten Thij, Tomasz Jakut, Tomek Wyrębowicz, Tommy Thorsen, Tony Ross, Tooru Fujisawa, Toru Kobayashi, Travis Leithead, Trevor Rowbotham, Trevor Saunders, Trey Eckels, triple soulignement, Tristan Fraipont, 保呂 毅 (Tsuyoshi Horo), Tyler Close, Valentin Gosu, Vardhan Gupta, Vas Sudanagunta, Veli Şenol, Victor Carbune, Victor Costan, Vipul Snehadeep Chawathe, Vitya Muhachev, Vlad Levin, Vladimir Katardjiev, Vladimir Vukićević, Vyacheslav Aristov, voracité, Walter Steiner, Wakaba, Wayne Carr, Wayne Pollock, Wellington Fernando de Macedo, Weston Ruter, Wilhelm Joys Andersen, Will Levine, Will Ray, William Chen, William Swanson, Willy Martin Aguirre Rodriguez, Wladimir Palant, Wojciech Mach, Wolfram Kriesing, Xan Gregg, xénothème, XhmikosR, Xida Chen, Xidorn Quan, Xue Fuqiao, Yang Chen, Yao Xiao, Yash Handa, Yay295, Ye-Kui Wang, Yehuda Katz, Yi Xu, Yi-An Huang, Yngve Nysaeter Pettersen, Yoav Weiss, Yonathan Randolph, Yu Huojiang, Yuki Okushi, Yury Delendik, 平野裕 (Yutaka Hirano), Yuzo Fujishima, Zhenbin Xu, 张智强 (Zhiqiang Zhang), Zoltan Herczeg et Øistein E Andersen, pour leurs commentaires utiles, petits et grands, qui ont conduit à des modifications de cette spécification au fil des ans.

Merci également à tous ceux qui ont déjà publié des articles sur HTML sur leurs blogs, leurs listes de diffusion publiques ou leurs forums, y compris tous les contributeurs aux [différentes listes W3C HTML WG](#) et aux [différentes listes WHATWG](#).

Un merci spécial à Richard Williamson pour avoir créé la première implémentation de `canvas` dans Safari, à partir de laquelle la fonctionnalité de canevas a été conçue.

Un merci spécial également aux employés de Microsoft qui ont été les premiers à implémenter le mécanisme de glisser-déposer basé sur les événements, `contenteditable` et d'autres fonctionnalités largement déployées pour la première fois par le navigateur Windows Internet Explorer.

Un merci spécial et 10 000 \$ à David Hyatt qui a proposé une implémentation cassée de l' [algorithme de l'agence d'adoption](#) que l'éditeur a dû désosser et corriger avant de l'utiliser dans la section d'analyse.

Merci aux participants à l'étude sur la convivialité des microdonnées de nous avoir permis d'utiliser leurs erreurs comme guide pour la conception de la fonction de microdonnées.

Merci aux nombreuses sources qui ont inspiré les exemples utilisés dans la spécification.

Merci également à la communauté des blogueurs Microsoft pour leurs idées, aux participants de l'atelier du W3C sur les applications Web et les documents composés pour leur inspiration, à l'équipe #mrt, à l'équipe #mrt.no et à l'équipe #whatwg, et à Pillar et Hedral pour leurs idées et leur soutien.

Merci à Igor Zhbanov d'avoir généré les versions PDF de la spécification.

Remerciements particuliers au [RICG](#) pour le développement de l'[picture](#) élément et des fonctionnalités associées ; en particulier grâce à Adrian Bateman, Bruce Lawson, David Newton, Ilya Grigorik, John Schoenick, Leon de Rijke, Mat Marquis, Marcos Cáceres, Tab Atkins, Theresa O'Connor et Yoav Weiss pour leurs contributions.

Un merci spécial au [WPWG](#) pour avoir incubé la fonctionnalité [des éléments personnalisés](#) . En particulier, merci à David Hyatt et Ian Hickson pour leur influence à travers les spécifications XBL, Dimitri Glazkov pour la première ébauche de la spécification des éléments personnalisés, et à Alex Komoroske, Alex Russell, Andres Rios, Boris Zbarsky, Brian Kardell, Daniel Buchner, Dominic Cooney, Erik Arvidsson, Elliott Sprehn, Hajime Morrita, Hayato Ito, Jan Miksovsky, Jonas Sicking, Olli Pettay, Rafael Weinstein, Roland Steiner, Ryosuke Niwa, Scott Miles, Steve Faulkner, Steve Orvell, Tab Atkins, Theresa O'Connor, Tim Perry et William Chen pour leurs contributions.

Remerciements particuliers au [CSSWG](#) pour le développement des [worklets](#) . En particulier, merci à Ian Kilpatrick pour son travail en tant qu'éditeur de la spécification originale des worklets.

Pendant une dizaine d'années à partir de 2003, cette norme a été presque entièrement rédigée par Ian Hickson ( [Google](#) , [ian@hixie.ch](#) ).

Depuis 2015, Simon Pieters ( [Mozilla](#) , [zcorpan@gmail.com](#) ), [Anne van Kesteren](#) ( [Apple](#) , [annevk@annevk.nl](#) ), [Philip Jägenstedt](#) ( [Google](#) , [philip@foolip.org](#) ) et [Domenic Denicola](#) ( [Google](#) , [d@domenic.me](#) ), tous contributeurs de longue date, se sont joints à Ian pour éditer directement le texte.

## Droits de propriété intellectuelle

L'image dans l'introduction est basée sur [une photo](#) de [Wonderlane](#) . ( [CC BY 2.0](#) )

L'image du loup dans l'introduction du contenu intégré est basée sur [une photo](#) de [Barry O'Neill](#) . ( [Domaine public](#) )

L'image de la balançoire kettlebell dans l'introduction du contenu intégré est basée sur [une photo](#) de [kokkarina](#) . ( [CC0 1.0](#) )

Le sprite Blue Robot Player utilisé dans la démo canvas est basé sur [une œuvre](#) de [JohnColburn](#) . ( [CC BY-SA 3.0](#) )

La photographie du robot 148 escaladant la tour du FIRST Robotics Competition 2013 Silicon Valley Regional est basée sur [une œuvre](#) de [Lenore Edman](#) . ( [CC BY 2.0](#) )

Le diagramme montrant comment `async` et `defer` impacter `script` le chargement est basé sur un diagramme similaire d' [un article de blog](#) de [Peter Beverloo](#) . ( [CC0 1.0](#) )

La démo de décodage d'image utilisée pour démontrer les travailleurs basés sur des modules s'appuie sur un exemple de code d' [un didacticiel](#) d' [Ilmari Heikkinen](#) . ( [CC BY 3.0](#) )

L' `<flag-icon>` exemple a été inspiré par [un élément personnalisé](#) de [Steven Skelton](#) . ( [MIT](#) )

Une partie de l'historique des révisions de l' `picture` élément et des fonctionnalités associées se trouve dans le [ResponsiveImagesCG/picture-element référentiel](#) , qui est disponible sous la [licence de logiciel et de document du W3C](#) .

Une partie de l' historique des révisions du `theme-color` nom des métadonnées se trouve dans le [whatwg/meta-theme-color référentiel](#) , qui est disponible sous [CC0](#) .

Une partie de l'historique des révisions de la fonctionnalité [des éléments personnalisés](#) se trouve dans le [w3c/webcomponents référentiel](#) , qui est disponible sous la [licence de logiciel et de document du W3C](#) .

Une partie de l'historique des révisions du `innerText` getter et du setter se trouve dans le [rocallahan/innerText-spec référentiel](#) , qui est disponible sous [CC0](#) .

Une partie de l' historique des révisions de la fonction [worklets](#) se trouve dans le [w3c/css-houdini-drafts référentiel](#) , qui est disponible sous la [licence de logiciel et de document W3C](#) .

Une partie de l' historique des révisions de la fonctionnalité [d' importation de cartes](#) se trouve dans le [WICG/import-maps référentiel](#) , qui est disponible sous la [licence de logiciel et de document W3C](#) .

Copyright © WHATWG (Apple, Google, Mozilla, Microsoft). Ce travail est sous licence [Creative Commons Attribution 4.0 International](#) . Dans la mesure où des parties de celui-ci sont incorporées dans le code source, ces parties du code source sont plutôt concédées sous la [licence BSD à 3 clauses](#) .

C'est le niveau de vie. Les personnes intéressées par la version d'examen des brevets devraient consulter le [projet d'examen du niveau de vie](#) .

## Table des matières

1. 1 Présentation
2. 2 Infrastructures communes
3. 3 Sémantique, structure et API des documents HTML
4. 4 Les éléments du HTML
5. 5 Microdonnées
6. 6 Interaction avec l'utilisateur
7. 7 Chargement des pages Web
8. 8 API d'applications Web
9. 9 Communications
10. 10 travailleurs du Web
11. 11 Worklets
12. 12 Stockage Web
13. 13 La syntaxe HTML
14. 14 La syntaxe XML
15. 15 Rendu
16. 16 fonctionnalités obsolètes
17. 17 Considérations relatives à l'IANA
18. Indice
19. Les références
20. Remerciements
21. Droits de propriété intellectuelle

## Table des matières complète

1. 1 Présentation
  1. 1.1 Où se situe cette spécification ?
  2. 1.2 Est-ce HTML5 ?
  3. 1.3 Contexte
  4. 1.4 Public
  5. 1.5 Portée
  6. 1.6 Historique
  7. 1.7 Notes de conception
    1. 1.7.1 Sérialisabilité de l'exécution du script
    2. 1.7.2 Conformité aux autres spécifications
    3. 1.7.3 Extensibilité
  8. 1.8 Syntaxe HTML vs XML

- 9. 1.9 Structure de cette spécification
  - 1. 1.9.1 Comment lire cette spécification
  - 2. 1.9.2 Conventions typographiques
- 10. 1.10 Une introduction rapide au HTML
  - 1. 1.10.1 Écrire des applications sécurisées avec HTML
  - 2. 1.10.2 Pièges courants à éviter lors de l'utilisation des API de script
  - 3. 1.10.3 Comment détecter les erreurs lors de l'écriture HTML : validateurs et vérificateurs de conformité
- 11. 1.11 Exigences de conformité pour les auteurs
  - 1. 1.11.1 Balisage de présentation
  - 2. 1.11.2 Erreurs de syntaxe
  - 3. 1.11.3 Restrictions sur les modèles de contenu et sur les valeurs d'attribut
- 12. 1.12 Lecture suggérée

## 2. 2 Infrastructures communes

- 1. 2.1 Terminologie
  - 1. 2.1.1 Parallélisme
  - 2. 2.1.2 Ressources
  - 3. 2.1.3 Compatibilité XML
  - 4. 2.1.4 Arbres DOM
  - 5. 2.1.5 Script
  - 6. 2.1.6 Plugins
  - 7. 2.1.7 Codages des caractères
  - 8. 2.1.8 Classes de conformité
  - 9. 2.1.9 Dépendances
  - 10. 2.1.10 Extensibilité
  - 11. 2.1.11 Interactions avec XPath et XSLT
- 2. 2.2 Fonctionnalités contrôlées par des politiques
- 3. 2.3 Microsyntaxes courantes
  - 1. 2.3.1 Idiomes d'analyseur courants
  - 2. 2.3.2 Attributs booléens
  - 3. 2.3.3 Mots clés et attributs énumérés
  - 4. 2.3.4 Numéros
    - 1. 2.3.4.1 Entiers signés

2. 2.3.4.2 Entiers non négatifs
    3. 2.3.4.3 Nombres à virgule flottante
    4. 2.3.4.4 Pourcentages et longueurs
    5. 2.3.4.5 Pourcentages et longueurs non nuls
    6. 2.3.4.6 Listes de nombres à virgule flottante
    7. 2.3.4.7 Listes de dimensions
  5. 2.3.5 Dates et heures
    1. 2.3.5.1 Mois
    2. 2.3.5.2 Dates
    3. 2.3.5.3 Dates sans année
    4. 2.3.5.4 Heures
    5. 2.3.5.5 Dates et heures locales
    6. 2.3.5.6 Fuseaux horaires
    7. 2.3.5.7 Dates et heures globales
    8. 2.3.5.8 Semaines
    9. 2.3.5.9 Durées
    10. 2.3.5.10 Moments plus vagues dans le temps
  6. 2.3.6 Couleurs
  7. 2.3.7 Jetons séparés par des espaces
  8. 2.3.8 Jetons séparés par des virgules
  9. 2.3.9 Références
  10. 2.3.10 Requêtes média
  11. 2.3.11 Valeurs internes uniques
4. 2.4 URL
  1. 2.4.1 Terminologie
  2. 2.4.2 Analyser les URL
  3. 2.4.3 Modifications dynamiques des URL de base
5. 2.5 Récupérer des ressources
  1. 2.5.1 Terminologie
  2. 2.5.2 Déterminer le type d'une ressource
  3. 2.5.3 Extraction des encodages de caractères des **meta**éléments
  4. 2.5.4 Attributs des paramètres CORS
  5. 2.5.5 Attributs de la politique de référence
  6. 2.5.6 Attributs nonce
  7. 2.5.7 Attributs de chargement différé
  8. 2.5.8 Attributs de blocage
  9. 2.5.9 Extraction des attributs de priorité
6. 2.6 Interfaces DOM communes
  1. 2.6.1 Refléter les attributs de contenu dans les attributs IDL

2. 2.6.2 Collectes
    1. 2.6.2.1 L' `HTMLAllCollection` interface
      1. 2.6.2.1.1 [[Appel]] ( *thisArgument* , *argumentsList* )
    2. 2.6.2.2 L' `HTMLFormControlsCollection` interface
    3. 2.6.2.3 L' `HTMLOptionsCollection` interface
  3. 2.6.3 L' `DOMStringList` interface
7. 2.7 Transmission sécurisée de données structurées
    1. 2.7.1 Objets sérialisables
    2. 2.7.2 Objets transférables
    3. 2.7.3 `StructuredSerializeInternal` ( *valeur* , *forStorage* [ , *mémoire* ] )
    4. 2.7.4 `StructuredSerialize` ( *valeur* )
    5. 2.7.5 `StructuredSerializeForStorage` ( *valeur* )
    6. 2.7.6 `StructuredDeserialize` ( *sérialisé* , *targetRealm* [ , *mémoire* ] )
    7. 2.7.7 `StructuredSerializeWithTransfer` ( *value* , *transferList* )
    8. 2.7.8 `StructuredDeserializeWithTransfer` ( *serializeWithTransferResult* , *targetRealm* )
    9. 2.7.9 Exécution de la sérialisation et transfert à partir d'autres spécifications
    10. 2.7.10 API de clonage structuré

### 3. 3 Sémantique, structure et API des documents HTML

1. 3.1 Documents
  1. 3.1.1 L' `Document` objet
  2. 3.1.2 L' `DocumentOrShadowRoot` interface
  3. 3.1.3 Gestion des métadonnées des ressources
  4. 3.1.4 Signaler l'état de chargement du document
  5. 3.1.5 Mécanisme de blocage du rendu
  6. 3.1.6 Accesseurs d'arborescence DOM
2. 3.2 Éléments
  1. 3.2.1 Sémantique
  2. 3.2.2 Éléments dans le DOM
  3. 3.2.3 Constructeurs d'éléments HTML
  4. 3.2.4 Définitions des éléments
    1. 3.2.4.1 Attributs
  5. 3.2.5 Modèles de contenu
    1. 3.2.5.1 Le modèle de contenu "rien"

2. 3.2.5.2 Types de contenu
  1. 3.2.5.2.1 Contenu des métadonnées
  2. 3.2.5.2.2 Contenu du flux
  3. 3.2.5.2.3 Sectionner le contenu
  4. 3.2.5.2.4 Contenu du titre
  5. 3.2.5.2.5 Phrase contenu
  6. 3.2.5.2.6 Contenu intégré
  7. 3.2.5.2.7 Contenu interactif
  8. 3.2.5.2.8 Contenu palpable
  9. 3.2.5.2.9 Éléments de support de script
3. 3.2.5.3 Modèles de contenu transparents
4. 3.2.5.4 Paragraphes
6. 3.2.6 Attributs globaux
  1. 3.2.6.1 L' `title`attribut
  2. 3.2.6.2 Les attributs `lang`et`xml:lang`
  3. 3.2.6.3 L' `translate`attribut
  4. 3.2.6.4 L' `dir`attribut
  5. 3.2.6.5 L' `style`attribut
  6. 3.2.6.6 Incorporation de données personnalisées non visibles avec les `data-*`attributs
7. 3.2.7 Les propriétés `innerText`et`outerText`
8. 3.2.8 Exigences relatives à l'algorithme bidirectionnel
  1. 3.2.8.1 Critères de conformité de création pour les caractères de formatage d'algorithme bidirectionnel
  2. 3.2.8.2 Critères de conformité de l'agent utilisateur
9. 3.2.9 Exigences liées à ARIA et aux API d'accessibilité de la plateforme

#### 4. 4 Les éléments du HTML

1. 4.1 L'élément document
  1. 4.1.1 L' `html`élément
2. 4.2 Métadonnées du document
  1. 4.2.1 L' `head`élément
  2. 4.2.2 L' `title`élément
  3. 4.2.3 L' `base`élément
  4. 4.2.4 L' `link`élément
    1. 4.2.4.1 Traitement de l' `media`attribut
    2. 4.2.4.2 Traitement de l' `type`attribut
    3. 4.2.4.3 Récupération et traitement d'une ressource à partir d'un `link`élément
    4. 4.2.4.4 Traitement `Link`des en-têtes ``



5. 4.2.4.5 Premiers indices
  6. 4.2.4.6 Fournir aux utilisateurs un moyen de suivre les hyperliens créés à l'aide de l' `link` élément
5. 4.2.5 L' `meta` élément
  1. 4.2.5.1 Noms de métadonnées standards
  2. 4.2.5.2 Autres noms de métadonnées
  3. 4.2.5.3 Directives pragmatiques
  4. 4.2.5.4 Spécification de l'encodage des caractères du document
6. 4.2.6 L' `style` élément
7. 4.2.7 Interactions du style et du script
3. 4.3 Rubriques
  1. 4.3.1 L' `body` élément
  2. 4.3.2 L' `article` élément
  3. 4.3.3 L' `section` élément
  4. 4.3.4 L' `nav` élément
  5. 4.3.5 L' `aside` élément
  6. 4.3.6 Les éléments `h1`, `h2`, `h3`, `h4`, `h5` et `h6`
  7. 4.3.7 L' `hgroup` élément
  8. 4.3.8 L' `header` élément
  9. 4.3.9 L' `footer` élément
  10. 4.3.10 L' `address` élément
  11. 4.3.11 Titres et contours
    1. 4.3.11.1 Exemples de contours
    2. 4.3.11.2 Exposer les contours aux utilisateurs
  12. 4.3.12 Résumé d'utilisation
    1. 4.3.12.1 Article ou rubrique ?
4. 4.4 Regroupement de contenu
  1. 4.4.1 L' `p` élément
  2. 4.4.2 L' `hr` élément
  3. 4.4.3 L' `pre` élément
  4. 4.4.4 L' `blockquote` élément
  5. 4.4.5 L' `ol` élément
  6. 4.4.6 L' `ul` élément
  7. 4.4.7 L' `menu` élément
  8. 4.4.8 L' `li` élément
  9. 4.4.9 L' `dl` élément
  10. 4.4.10 L' `dt` élément

- 11.4.4.11 L' `dd`élément
- 12.4.4.12 L' `figure`élément
- 13.4.4.13 L' `figcaption`élément
- 14.4.4.14 L' `main`élément
- 15.4.4.15 L' `div`élément

## 5. 4.5 Sémantique au niveau du texte

- 1. 4.5.1 L' `a`élément
- 2. 4.5.2 L' `em`élément
- 3. 4.5.3 L' `strong`élément
- 4. 4.5.4 L' `small`élément
- 5. 4.5.5 L' `s`élément
- 6. 4.5.6 L' `cite`élément
- 7. 4.5.7 L' `q`élément
- 8. 4.5.8 L' `dfn`élément
- 9. 4.5.9 L' `abbr`élément
- 10.4.5.10 L' `ruby`élément
- 11.4.5.11 L' `rt`élément
- 12.4.5.12 L' `rp`élément
- 13.4.5.13 L' `data`élément
- 14.4.5.14 L' `time`élément
- 15.4.5.15 L' `code`élément
- 16.4.5.16 L' `var`élément
- 17.4.5.17 L' `samp`élément
- 18.4.5.18 L' `kbd`élément
- 19.4.5.19 Les éléments `sub``et``sup`
- 20.4.5.20 L' `i`élément
- 21.4.5.21 L' `b`élément
- 22.4.5.22 L' `u`élément
- 23.4.5.23 L' `mark`élément
- 24.4.5.24 L' `bdi`élément
- 25.4.5.25 L' `bdo`élément
- 26.4.5.26 L' `span`élément
- 27.4.5.27 L' `br`élément
- 28.4.5.28 L' `wbr`élément
- 29.4.5.29 Résumé d'utilisation

## 6. 4.6 Liens

1. 4.6.1 Présentation
2. 4.6.2 Liens créés par `aet` `area` éléments
3. 4.6.3 API pour `aet` `area` éléments
4. 4.6.4 Suivre des hyperliens
5. 4.6.5 Téléchargement des ressources
6. 4.6.6 Audit des hyperliens
  1. 4.6.6.1 Les en-têtes `` Ping-From` et `` Ping-To``
7. 4.6.7 Types de lien
  1. 4.6.7.1 Type de lien `" alternate"`
  2. 4.6.7.2 Type de lien `" author"`
  3. 4.6.7.3 Type de lien `" bookmark"`
  4. 4.6.7.4 Type de lien `" canonical"`
  5. 4.6.7.5 Type de lien `" dns-prefetch"`
  6. 4.6.7.6 Type de lien `" external"`
  7. 4.6.7.7 Type de lien `" help"`
  8. 4.6.7.8 Type de lien `" icon"`
  9. 4.6.7.9 Type de lien `" license"`
  10. 4.6.7.10 Type de lien `" manifest"`
  11. 4.6.7.11 Type de lien `" modulepreload"`
  12. 4.6.7.12 Type de lien `" nofollow"`
  13. 4.6.7.13 Type de lien `" noopener"`
  14. 4.6.7.14 Type de lien `" norereferrer"`
  15. 4.6.7.15 Type de lien `" opener"`
  16. 4.6.7.16 Type de lien `" pingback"`
  17. 4.6.7.17 Type de lien `" preconnect"`
  18. 4.6.7.18 Type de lien `" prefetch"`
  19. 4.6.7.19 Type de lien `" preload"`
  20. 4.6.7.20 Type de lien `" prerender"`
  21. 4.6.7.21 Type de lien `" search"`
  22. 4.6.7.22 Type de lien `" stylesheet"`
  23. 4.6.7.23 Type de lien `" tag"`
  24. 4.6.7.24 Types de liens séquentiels
    1. 4.6.7.24.1 Type de lien `" next"`
    2. 4.6.7.24.2 Type de lien `" prev"`
  25. 4.6.7.25 Autres types de liens
7. 4.7 Modifications
  1. 4.7.1 L' `ins` élément
  2. 4.7.2 L' `del` élément
  3. 4.7.3 Attributs communs aux éléments `ins` `del`
  4. 4.7.4 Modifications et paragraphes

5. 4.7.5 Modifications et listes
6. 4.7.6 Modifications et tableaux
8. 4.8 Contenu intégré
  1. 4.8.1 L' `picture` élément
  2. 4.8.2 L' `source` élément
  3. 4.8.3 L' `img` élément
  4. 4.8.4 Images
    1. 4.8.4.1 Présentation
      1. 4.8.4.1.1 Images adaptatives
    2. 4.8.4.2 Attributs communs aux éléments `source`, `img` et `link`
      1. 4.8.4.2.1 Attributs Srcset
      2. 4.8.4.2.2 Attributs de tailles
    3. 4.8.4.3 Modèle de traitement
      1. 4.8.4.3.1 Quand obtenir des images
      2. 4.8.4.3.2 Réagir aux mutations du DOM
      3. 4.8.4.3.3 La liste des images disponibles
      4. 4.8.4.3.4 Décodage des images
      5. 4.8.4.3.5 Mise à jour des données d'image
      6. 4.8.4.3.6 Préparation d'une image pour la présentation
      7. 4.8.4.3.7 Sélection d'une source d'images
      8. 4.8.4.3.8 Création d'un ensemble source à partir d'attributs
      9. 4.8.4.3.9 Mise à jour du poste source
      10. 4.8.4.3.10 Analyse d'un attribut srcset
      11. 4.8.4.3.11 Analyse d'un attribut tailles
      12. 4.8.4.3.12 Normalisation des densités de source
      13. 4.8.4.3.13 Réagir aux changements d'environnement
    4. 4.8.4.4 Exigences relatives à la fourniture de texte pour remplacer les images
      1. 4.8.4.4.1 Directives générales
      2. 4.8.4.4.2 Un lien ou un bouton ne contenant que l'image
      3. 4.8.4.4.3 Une phrase ou un paragraphe avec une représentation graphique alternative : diagrammes, diagrammes, graphiques, cartes, illustrations
      4. 4.8.4.4.4 Une courte phrase ou une étiquette avec une représentation graphique alternative : icônes, logos
      5. 4.8.4.4.5 Texte rendu dans un graphique pour un effet typographique
      6. 4.8.4.4.6 Une représentation graphique d'une partie du texte environnant
      7. 4.8.4.4.7 Images auxiliaires

8. 4.8.4.4.8 Une image purement décorative qui n'ajoute aucune information
9. 4.8.4.4.9 Un groupe d'images qui forment une seule image plus grande sans liens
10. 4.8.4.4.10 Un groupe d'images qui forment une seule image plus grande avec des liens
11. 4.8.4.4.11 Un élément clé du contenu
12. 4.8.4.4.12 Une image non destinée à l'utilisateur
13. 4.8.4.4.13 Une image dans un e-mail ou un document privé destiné à une personne spécifique connue pour être en mesure de visualiser des images
14. 4.8.4.4.14 Conseils pour les générateurs de balisage
15. 4.8.4.4.15 Conseils pour les vérificateurs de conformité
5. 4.8.5 L' `iframe` élément
6. 4.8.6 L' `embed` élément
7. 4.8.7 L' `object` élément
8. 4.8.8 L' `video` élément
9. 4.8.9 L' `audio` élément
10. 4.8.10 L' `track` élément
11. 4.8.11 Éléments multimédias
  1. 4.8.11.1 Codes d'erreur
  2. 4.8.11.2 Emplacement de la ressource média
  3. 4.8.11.3 Type MIME
  4. 4.8.11.4 États du réseau
  5. 4.8.11.5 Chargement de la ressource média
  6. 4.8.11.6 Décalages dans la ressource média
  7. 4.8.11.7 États prêts
  8. 4.8.11.8 Lecture de la ressource multimédia
  9. 4.8.11.9 Recherche
  10. 4.8.11.10 Ressources multimédias avec plusieurs pistes multimédias
    1. 4.8.11.10.1 `AudioTrackList` et `VideoTrackList` objets
    2. 4.8.11.10.2 Sélection de pistes audio et vidéo spécifiques de manière déclarative
  11. 4.8.11.11 Pistes de texte chronométrées
    1. 4.8.11.11.1 Modèle de piste de texte
    2. 4.8.11.11.2 Recherche de pistes de texte dans la bande
    3. 4.8.11.11.3 Recherche de pistes de texte hors bande

4. 4.8.11.11.4 Directives pour exposer des repères dans divers formats en tant que repères de piste de texte
  5. 4.8.11.11.5 API de suivi de texte
  6. 4.8.11.11.6 Gestionnaires d'événements pour les objets des API de suivi de texte
  7. 4.8.11.11.7 Meilleures pratiques pour les pistes de texte de métadonnées
  12. 4.8.11.12 Identification d'un type de piste via une URL
  13. 4.8.11.13 Interface utilisateur
  14. 4.8.11.14 Plages horaires
  15. 4.8.11.15 L' `TrackEvent` interface
  16. 4.8.11.16 Résumé des événements
  17. 4.8.11.17 Considérations relatives à la sécurité et à la confidentialité
  18. 4.8.11.18 Meilleures pratiques pour les auteurs utilisant des éléments multimédias
  19. 4.8.11.19 Meilleures pratiques pour les implémenteurs d'éléments multimédias
12. 4.8.12 L' `map` élément
13. 4.8.13 L' `area` élément
14. 4.8.14 Images cliquables
  1. 4.8.14.1 Création
  2. 4.8.14.2 Modèle de traitement
15. 4.8.15 MathML
16. 4.8.16 SVG
17. 4.8.17 Attributs de dimension
9. 4.9 Données tabulaires
  1. 4.9.1 L' `table` élément
    1. 4.9.1.1 Techniques de description des tableaux
    2. 4.9.1.2 Techniques de conception de table
  2. 4.9.2 L' `caption` élément
  3. 4.9.3 L' `colgroup` élément
  4. 4.9.4 L' `col` élément
  5. 4.9.5 L' `tbody` élément
  6. 4.9.6 L' `thead` élément
  7. 4.9.7 L' `tfoot` élément
  8. 4.9.8 L' `tr` élément
  9. 4.9.9 L' `td` élément
  10. 4.9.10 L' `th` élément
  11. 4.9.11 Attributs communs aux éléments `td` et `th`

#### 12.4.9.12 Modèle de traitement

1. 4.9.12.1 Formation d'un tableau
2. 4.9.12.2 Formation de relations entre les cellules de données et les cellules d'en-tête

#### 13.4.9.13 Exemples

### 10.4.10 Formulaires

#### 1. 4.10.1 Présentation

1. 4.10.1.1 Ecrire l'interface utilisateur d'un formulaire
2. 4.10.1.2 Implémentation du traitement côté serveur pour un formulaire
3. 4.10.1.3 Configurer un formulaire pour communiquer avec un serveur
4. 4.10.1.4 Validation du formulaire côté client
5. 4.10.1.5 Activation du remplissage automatique côté client des contrôles de formulaire
6. 4.10.1.6 Améliorer l'expérience utilisateur sur les appareils mobiles
7. 4.10.1.7 La différence entre le type de champ, le nom du champ de remplissage automatique et la modalité de saisie
8. 4.10.1.8 Formats de date, d'heure et de nombre

#### 2. 4.10.2 Catégories

#### 3. 4.10.3 L' `form` élément

#### 4. 4.10.4 L' `label` élément

#### 5. 4.10.5 L' `input` élément

##### 1. 4.10.5.1 Etats de l' `type` attribut

1. 4.10.5.1.1 État masqué ( `type=hidden` )
2. 4.10.5.1.2 État Texte ( `type=text` ) et État Recherche ( `type=search` )
3. 4.10.5.1.3 État du téléphone ( `type=tel` )
4. 4.10.5.1.4 État de l'URL ( `type=url` )
5. 4.10.5.1.5 État de l'e-mail ( `type=email` )
6. 4.10.5.1.6 État du mot de passe ( `type=password` )
7. 4.10.5.1.7 État de la date ( `type=date` )
8. 4.10.5.1.8 État du mois ( `type=month` )
9. 4.10.5.1.9 État de la semaine ( `type=week` )
10. 4.10.5.1.10 État horaire ( `type=time` )
11. 4.10.5.1.11 État de la date et de l'heure locales ( `type=datetime-local` )
12. 4.10.5.1.12 État du nombre ( `type=number` )
13. 4.10.5.1.13 État de la plage ( `type=range` )
14. 4.10.5.1.14 État de la couleur ( `type=color` )
15. 4.10.5.1.15 État de la case à cocher ( `type=checkbox` )
16. 4.10.5.1.16 État du bouton radio ( `type=radio` )

- 17.4.10.5.1.17 État de téléchargement de fichier  
( `type=file`)
- 18.4.10.5.1.18 État du bouton Soumettre  
( `type=submit`)
- 19.4.10.5.1.19 État du bouton d'image ( `type=image`)
- 20.4.10.5.1.20 État du bouton de réinitialisation  
( `type=reset`)
- 21.4.10.5.1.21 État du bouton ( `type=button`)
- 2. 4.10.5.2 Notes d'implémentation concernant la localisation des contrôles de formulaire
- 3. 4.10.5.3 `input` Attributs communs des éléments
  - 1. 4.10.5.3.1 Les attributs `maxlength` et `minlength`
  - 2. 4.10.5.3.2 L' `size`attribut
  - 3. 4.10.5.3.3 L' `readonly`attribut
  - 4. 4.10.5.3.4 L' `required`attribut
  - 5. 4.10.5.3.5 L' `multiple`attribut
  - 6. 4.10.5.3.6 L' `pattern`attribut
  - 7. 4.10.5.3.7 Les attributs `min` et `max`
  - 8. 4.10.5.3.8 L' `step`attribut
  - 9. 4.10.5.3.9 L' `list`attribut
  - 10.4.10.5.3.10 L' `placeholder`attribut
- 4. 4.10.5.4 `input` API d'éléments communs
- 5. 4.10.5.5 Comportements d'événements courants
- 6. 4.10.6 L' `button`élément
- 7. 4.10.7 L' `select`élément
- 8. 4.10.8 L' `datalist`élément
- 9. 4.10.9 L' `optgroup`élément
- 10.4.10.10 L' `option`élément
- 11.4.10.11 L' `textarea`élément
- 12.4.10.12 L' `output`élément
- 13.4.10.13 L' `progress`élément
- 14.4.10.14 L' `meter`élément
- 15.4.10.15 L' `fieldset`élément
- 16.4.10.16 L' `legend`élément
- 17.4.10.17 Infrastructure de contrôle des formulaires
  - 1. 4.10.17.1 La valeur d'un contrôle de formulaire
  - 2. 4.10.17.2 Mutabilité
  - 3. 4.10.17.3 Association des champs et des formulaires
- 18.4.10.18 Attributs communs aux champs de formulaire
  - 1. 4.10.18.1 Nommer les champs de formulaire :  
l' `name`attribut
  - 2. 4.10.18.2 Soumission de la directionnalité de l'élément :  
l' `dirname`attribut



3. 4.10.18.3 Limitation de la longueur des entrées utilisateur : l' `maxlength`attribut
4. 4.10.18.4 Définition des exigences de longueur d'entrée minimale : l' `minlength`attribut
5. 4.10.18.5 Activation et désactivation des contrôles de formulaire : l' `disabled`attribut
6. 4.10.18.6 Attributs de soumission de formulaire
7. 4.10.18.7 Remplissage automatique
  1. 4.10.18.7.1 Champs de remplissage automatique : l' `autocomplete`attribut
  2. 4.10.18.7.2 Modèle de traitement
- 19.4.10.19 API pour les sélections de contrôle de texte
- 20.4.10.20 Contraintes
  1. 4.10.20.1 Définitions
  2. 4.10.20.2 Validation des contraintes
  3. 4.10.20.3 L'API de validation de contraintes
  4. 4.10.20.4 Sécurité
- 21.4.10.21 Soumission du formulaire
  1. 4.10.21.1 Présentation
  2. 4.10.21.2 Soumission implicite
  3. 4.10.21.3 Algorithme de soumission de formulaire
  4. 4.10.21.4 Construire la liste des inscrits
  5. 4.10.21.5 Sélection d'un encodage de soumission de formulaire
  6. 4.10.21.6 Conversion d'une liste d'entrées en une liste de paires nom-valeur
  7. 4.10.21.7 Données de formulaire codées en URL
  8. 4.10.21.8 Données de formulaire en plusieurs parties
  9. 4.10.21.9 Données de formulaire en texte brut
  - 10.4.10.21.10 L' `SubmitEvent`interface
  - 11.4.10.21.11 L' `FormDataEvent`interface
- 22.4.10.22 Réinitialiser un formulaire
- 11.4.11 Éléments interactifs
  1. 4.11.1 L' `details`élément
  2. 4.11.2 L' `summary`élément
  3. 4.11.3 Commandes
    1. 4.11.3.1 Facettes
    2. 4.11.3.2 Utilisation de l' `a`élément pour définir une commande
    3. 4.11.3.3 Utilisation de l' `button`élément pour définir une commande
    4. 4.11.3.4 Utilisation de l' `input`élément pour définir une commande

5. 4.11.3.5 Utilisation de l' `option` élément pour définir une commande
  6. 4.11.3.6 Utilisation de l' `accesskey` attribut sur un `legend` élément pour définir une commande
  7. 4.11.3.7 Utilisation de l' `accesskey` attribut pour définir une commande sur d'autres éléments
  4. 4.11.4 L' `dialog` élément
- 12.4.12 Script

1. 4.12.1 L' `script` élément
  1. 4.12.1.1 Modèle de traitement
  2. 4.12.1.2 Langages de script
  3. 4.12.1.3 Restrictions pour le contenu des `script` éléments
  4. 4.12.1.4 Documentation en ligne pour les scripts externes
  5. 4.12.1.5 Interaction des `script` éléments et XSLT
2. 4.12.2 L' `noscript` élément
3. 4.12.3 L' `template` élément
  1. 4.12.3.1 Interaction des `template` éléments avec XSLT et XPath
4. 4.12.4 L' `slot` élément
5. 4.12.5 L' `canvas` élément
  1. 4.12.5.1 Le contexte de rendu 2D
    1. 4.12.5.1.1 Remarques sur la mise en œuvre
    2. 4.12.5.1.2 L'état du canevas
    3. 4.12.5.1.3 Styles de ligne
    4. 4.12.5.1.4 Styles de texte
    5. 4.12.5.1.5 Construire des chemins
    6. 4.12.5.1.6 `Path2D` objets
    7. 4.12.5.1.7 Transformations
    8. 4.12.5.1.8 Sources d'images pour les contextes de rendu 2D
    9. 4.12.5.1.9 Styles de remplissage et de trait
    10. 4.12.5.1.10 Dessiner des rectangles sur le bitmap
    11. 4.12.5.1.11 Dessiner du texte sur le bitmap
    12. 4.12.5.1.12 Dessiner des chemins vers le canevas
    13. 4.12.5.1.13 Dessiner des anneaux de mise au point et des chemins de défilement dans la vue
    14. 4.12.5.1.14 Dessiner des images
    15. 4.12.5.1.15 Manipulation des pixels
    16. 4.12.5.1.16 Composition
    17. 4.12.5.1.17 Lissage des images
    18. 4.12.5.1.18 Ombres
    19. 4.12.5.1.19 Filtres
    20. 4.12.5.1.20 Utilisation de filtres SVG définis en externe
    21. 4.12.5.1.21 Modèle de dessin

- 22.4.12.5.1.22 Bonnes pratiques
    - 23.4.12.5.1.23 Exemples
  - 2. 4.12.5.2 Le `ImageBitmap` contexte de rendu
    - 1. 4.12.5.2.1 Présentation
    - 2. 4.12.5.2.2 L' `ImageBitmapRenderingContext` interface
  - 3. 4.12.5.3 L' `OffscreenCanvas` interface
    - 1. 4.12.5.3.1 Le contexte de rendu 2D hors écran
  - 4. 4.12.5.4 Espaces colorimétriques et conversion d'espace colorimétrique
  - 5. 4.12.5.5 Sérialisation de bitmaps dans un fichier
  - 6. 4.12.5.6 Sécurité avec `canvas` éléments
  - 7. 4.12.5.7 Alpha prémultiplié et contexte de rendu 2D
- 13.4.13 Éléments personnalisés
- 1. 4.13.1 Présentation
    - 1. 4.13.1.1 Créer un élément personnalisé autonome
    - 2. 4.13.1.2 Création d'un élément personnalisé associé à un formulaire
    - 3. 4.13.1.3 Création d'un élément personnalisé avec des rôles, des états et des propriétés accessibles par défaut
    - 4. 4.13.1.4 Création d'un élément intégré personnalisé
    - 5. 4.13.1.5 Inconvénients des éléments personnalisés autonomes
    - 6. 4.13.1.6 Mettre à jour les éléments après leur création
  - 2. 4.13.2 Exigences pour les constructeurs d'éléments personnalisés et les réactions
  - 3. 4.13.3 Concepts de base
  - 4. 4.13.4 L' `CustomElementRegistry` interface
  - 5. 4.13.5 Mises à jour
  - 6. 4.13.6 Réactions d'élément personnalisées
  - 7. 4.13.7 Éléments internes
    - 1. 4.13.7.1 L' `ElementInternals` interface
    - 2. 4.13.7.2 Accès racine fantôme
    - 3. 4.13.7.3 Éléments personnalisés associés au formulaire
    - 4. 4.13.7.4 Sémantique d'accessibilité
- 14.4.14 Idiomes communs sans éléments dédiés
- 1. 4.14.1 Navigation fil d'Ariane
  - 2. 4.14.2 Nuages de tags
  - 3. 4.14.3 Conversation
  - 4. 4.14.4 Notes de bas de page
- 15.4.15 Éléments désactivés

#### 16.4.16 Correspondance d'éléments HTML à l'aide de sélecteurs et de CSS

1. 4.16.1 Sensibilité à la casse de la fonction CSS 'attr()'
2. 4.16.2 Sensibilité à la casse des sélecteurs
3. 4.16.3 Pseudo-classes

### 5. 5 Microdonnées

#### 1. 5.1 Présentation

1. 5.1.1 Aperçu
2. 5.1.2 La syntaxe de base
3. 5.1.3 Éléments typés
4. 5.1.4 Identifiants globaux pour les éléments
5. 5.1.5 Sélection de noms lors de la définition de vocabulaires

#### 2. 5.2 Encodage des microdonnées

1. 5.2.1 Le modèle de microdonnées
2. 5.2.2 Articles
3. 5.2.3 Noms : l' `itemprop` attribut
4. 5.2.4 Valeurs
5. 5.2.5 Associer des noms à des éléments
6. 5.2.6 Microdonnées et autres espaces de noms

#### 3. 5.3 Exemples de vocabulaires de microdonnées

1. 5.3.1 vCard
  1. 5.3.1.1 Conversion en vCard
  2. 5.3.1.2 Exemples
2. 5.3.2 vÉvénement
  1. 5.3.2.1 Conversion vers iCalendar
  2. 5.3.2.2 Exemples
3. 5.3.3 Autorisation des travaux
  1. 5.3.3.1 Exemples

#### 4. 5.4 Conversion de HTML vers d'autres formats

1. 5.4.1 JSON

### 6. 6 Interaction avec l'utilisateur

1. 6.1 L' `hidden` attribut
2. 6.2 Visibilité des pages

3. 6.3 Sous-arbres inertes
  1. 6.3.1 Dialogues modaux et sous-arbres inertes
  2. 6.3.2 L' `inert`attribut
4. 6.4 Suivi de l'activation de l'utilisateur
  1. 6.4.1 Modèle de données
  2. 6.4.2 Modèle de traitement
  3. 6.4.3 API fermées par l'activation de l'utilisateur
  4. 6.4.4 L' `UserActivation`interface
5. 6.5 Comportement d'activation des éléments
6. 6.6 Mise au point
  1. 6.6.1 Présentation
  2. 6.6.2 Modèle de données
  3. 6.6.3 L' `tabindex`attribut
  4. 6.6.4 Modèle de traitement
  5. 6.6.5 Navigation de mise au point séquentielle
  6. 6.6.6 API de gestion des focus
  7. 6.6.7 L' `autofocus`attribut
7. 6.7 Attribuer des raccourcis clavier
  1. 6.7.1 Présentation
  2. 6.7.2 L' `accesskey` attribut
  3. 6.7.3 Modèle de traitement
8. 6.8 Édition
  1. 6.8.1 Rendre les régions de document modifiables : l' `contenteditable`attribut de contenu
  2. 6.8.2 Rendre des documents entiers modifiables : le `designMode`getter et le setter
  3. 6.8.3 Meilleures pratiques pour les éditeurs de pages
  4. 6.8.4 Modification des API
  5. 6.8.5 Vérification orthographique et grammaticale
  6. 6.8.6 Autocapitalisation
  7. 6.8.7 Modalités de saisie : l' `inputmode`attribut
  8. 6.8.8 Modalités de saisie : l' `enterkeyhint` attribut
9. 6.9 Rechercher dans la page
  1. 6.9.1 Présentation
  2. 6.9.2 Interaction avec `details``sethidden=until-found`
  3. 6.9.3 Interaction avec la sélection
- 10.6.10 Glisser-déposer

1. 6.10.1 Présentation
2. 6.10.2 Le magasin de données de glissement
3. 6.10.3 L' `DataTransfer` interface
  1. 6.10.3.1 L' `DataTransferItemList` interface
  2. 6.10.3.2 L' `DataTransferItem` interface
4. 6.10.4 L' `DragEvent` interface
5. 6.10.5 Modèle de traitement
6. 6.10.6 Résumé des événements
7. 6.10.7 L' `draggable` attribut
8. 6.10.8 Risques de sécurité dans le modèle glisser-déposer
- 11.6.11 L' `popover` attribut
  1. 6.11.1 Les attributs de la cible popover
  2. 6.11.2 Ignorer la lumière Popover
  3. 6.11.3 L' `ToggleEvent` interface

## 7. 7 Chargement des pages Web

1. 7.1 Notions complémentaires
  1. 7.1.1 Origines
    1. 7.1.1.1 Sites
    2. 7.1.1.2 Assouplissement de la restriction de même origine
  2. 7.1.2 Grappes d'agents à clé d'origine
  3. 7.1.3 Politiques d'ouverture d'origine croisée
    1. 7.1.3.1 Les en-têtes
    2. 7.1.3.2 Changements de groupe de contexte de navigation en raison de la politique d'ouverture d'origine croisée
    3. 7.1.3.3 Rapports
  4. 7.1.4 Politiques d'intégration inter-origines
    1. 7.1.4.1 Les en-têtes
    2. 7.1.4.2 Contrôles de politique d'incorporation
  5. 7.1.5 Bac à sable
  6. 7.1.6 Conteneurs de politique
2. 7.2 API liées à la navigation et à l'historique des sessions
  1. 7.2.1 Infrastructure de sécurité pour les objets `Window`, `WindowProxy` et `Location`
    1. 7.2.1.1 Intégration avec IDL
    2. 7.2.1.2 Emplacement interne partagé : `[[CrossOriginPropertyDescriptorMap]]`

3. 7.2.1.3 Opérations abstraites partagées
  1. 7.2.1.3.1 CrossOriginProperties ( *O* )
  2. 7.2.1.3.2 CrossOriginPropertyFallback ( *P* )
  3. 7.2.1.3.3 IsPlatformObjectSameOrigin ( *O* )
  4. 7.2.1.3.4 CrossOriginGetOwnPropertyHelper ( *O* , *P* )
  5. 7.2.1.3.5 CrossOriginGet ( *O* , *P* , Récepteur )
  6. 7.2.1.3.6 CrossOriginSet ( *O* , *P* , *V* , Récepteur )
  7. 7.2.1.3.7 CrossOriginOwnPropertyKeys ( *O* )
2. 7.2.2 L' *Window* objet
  1. 7.2.2.1 Ouverture et fermeture des fenêtres
  2. 7.2.2.2 Accès indexé sur l' *Window* objet
  3. 7.2.2.3 Accès nommé sur l' *Window* objet
  4. 7.2.2.4 Accéder aux fenêtres associées
  5. 7.2.2.5 API des éléments d'interface de navigateur historiques
  6. 7.2.2.6 Paramètres de script pour *Window* les objets
3. 7.2.3 L' *WindowProxy* objet exotique
  1. 7.2.3.1 [[GetPrototypeOf]] ( )
  2. 7.2.3.2 [[SetPrototypeOf]] ( *V* )
  3. 7.2.3.3 [[EstExtensible]] ( )
  4. 7.2.3.4 [[Empêcher les extensions]] ( )
  5. 7.2.3.5 [[GetOwnProperty]] ( *P* )
  6. 7.2.3.6 [[DéfinirProperty]] ( *P* , *Desc* )
  7. 7.2.3.7 [[Obtenir]] ( *P* , Récepteur )
  8. 7.2.3.8 [[Régler]] ( *P* , *V* , Récepteur )
  9. 7.2.3.9 [[Supprimer]] ( *P* )
  10. 7.2.3.10 [[OwnPropertyKeys]] ( )
4. 7.2.4 L' *Location* interface
  1. 7.2.4.1 [[GetPrototypeOf]] ( )
  2. 7.2.4.2 [[SetPrototypeOf]] ( *V* )
  3. 7.2.4.3 [[EstExtensible]] ( )
  4. 7.2.4.4 [[Empêcher les extensions]] ( )
  5. 7.2.4.5 [[GetOwnProperty]] ( *P* )
  6. 7.2.4.6 [[DéfinirProperty]] ( *P* , *Desc* )
  7. 7.2.4.7 [[Obtenir]] ( *P* , Récepteur )
  8. 7.2.4.8 [[Régler]] ( *P* , *V* , Récepteur )
  9. 7.2.4.9 [[Supprimer]] ( *P* )
  10. 7.2.4.10 [[OwnPropertyKeys]] ( )
5. 7.2.5 L' *History* interface
6. 7.2.6 Interfaces d'événements
  1. 7.2.6.1 L' *PopStateEvent* interface
  2. 7.2.6.2 L' *HashChangeEvent* interface

3. 7.2.6.3 L' `PageTransitionEvent` interface
  4. 7.2.6.4 L' `BeforeUnloadEvent` interface
3. 7.3 Infrastructure pour les séquences de documents
  1. 7.3.1 Navigables
    1. 7.3.1.1 Navigables traversables
    2. 7.3.1.2 Traversables de niveau supérieur
    3. 7.3.1.3 Navigables enfants
    4. 7.3.1.4 Diagrammes de Jake
    5. 7.3.1.5 Collections navigables associées
    6. 7.3.1.6 Destruction navigable
    7. 7.3.1.7 Noms de cibles navigables
  2. 7.3.2 Contextes de navigation
    1. 7.3.2.1 Création de contextes de navigation
    2. 7.3.2.2 Contextes de navigation associés
    3. 7.3.2.3 Regroupements de contextes de navigation
  3. 7.3.3 Documents pleinement actifs
4. 7.4 Navigation et historique des sessions
  1. 7.4.1 Historique des sessions
    1. 7.4.1.1 Entrées de l'historique des sessions
    2. 7.4.1.2 État des documents
    3. 7.4.1.3 Modifications centralisées de l'historique des sessions
    4. 7.4.1.4 Opérations de bas niveau sur l'historique des sessions
  2. 7.4.2 Navigation
    1. 7.4.2.1 Notions complémentaires
    2. 7.4.2.2 Commencer la navigation
    3. 7.4.2.3 Terminer la navigation
      1. 7.4.2.3.1 Le cas habituel de navigation entre documents
      2. 7.4.2.3.2 Le `javascript:` cas particulier de l'URL
      3. 7.4.2.3.3 Navigations fragmentaires
      4. 7.4.2.3.4 Schémas de non-extraction et logiciels externes
    4. 7.4.2.4 Empêcher la navigation
  3. 7.4.3 Rechargement et déplacement
  4. 7.4.4 "Navigations" synchrones sans fragment
  5. 7.4.5 Remplir une entrée d'historique de session
  6. 7.4.6 Application de l'étape d'historique
    1. 7.4.6.1 Mise à jour du traversable
    2. 7.4.6.2 Mise à jour du document
    3. 7.4.6.3 Défilement vers un fragment



4. 7.4.6.4 État d'entrée d'historique persistant
  5. 7.5 Cycle de vie des documents
    1. 7.5.1 Infrastructure de création de documents partagés
    2. 7.5.2 Chargement de documents HTML
    3. 7.5.3 Chargement de documents XML
    4. 7.5.4 Chargement de documents texte
    5. 7.5.5 Chargement `multipart/x-mixed-replace` de documents
    6. 7.5.6 Chargement de documents multimédias
    7. 7.5.7 Chargement d'un document pour un contenu en ligne qui n'a pas de DOM
    8. 7.5.8 Terminer le processus de chargement
    9. 7.5.9 Déchargement des documents
    10. 7.5.10 Destruction de documents
    11. 7.5.11 Abandon du chargement d'un document
  6. 7.6 L' `X-Frame-Options` en-tête ``
  7. 7.7 L' `Refresh` en-tête ``
  8. 7.8 Considérations sur l'interface utilisateur du navigateur
8. 8 API d'applications Web
1. 8.1 Script
    1. 8.1.1 Présentation
    2. 8.1.2 Agents et grappes d'agents
      1. 8.1.2.1 Intégration avec le formalisme de l'agent JavaScript
      2. 8.1.2.2 Intégration avec le formalisme du cluster d'agents JavaScript
    3. 8.1.3 Les royaumes et leurs homologues
      1. 8.1.3.1 Environnements
      2. 8.1.3.2 Objets de paramètres d'environnement
      3. 8.1.3.3 Domaines, objets de paramètres et objets globaux
        1. 8.1.3.3.1 Saisie
        2. 8.1.3.3.2 Titulaire
        3. 8.1.3.3.3 Courant
        4. 8.1.3.3.4 Pertinent
      4. 8.1.3.4 Activation et désactivation des scripts
      5. 8.1.3.5 Contextes sécurisés
    4. 8.1.4 Modèle de traitement de script
      1. 8.1.4.1 Scénarios
      2. 8.1.4.2 Récupérer des scripts

3. 8.1.4.3 Création de scripts
4. 8.1.4.4 Appel de scripts
5. 8.1.4.5 Arrêter des scripts
6. 8.1.4.6 Erreurs de script d'exécution
7. 8.1.4.7 Refus de promesses non gérées
8. 8.1.4.8 Importer les résultats d'analyse de carte
5. 8.1.5 Résolution du spécificateur de module
  1. 8.1.5.1 L'algorithme de résolution
  2. 8.1.5.2 Importer des cartes
  3. 8.1.5.3 Importer le modèle de traitement de carte
6. 8.1.6 Crochets d'hôte de spécification JavaScript
  1. 8.1.6.1 HostEnsureCanAddPrivateElement( *O* )
  2. 8.1.6.2 HostEnsureCanCompileStrings( *domaine* )
  3. 8.1.6.3 HostPromiseRejectionTracker( *promesse* , *opération* )
  4. 8.1.6.4 Crochets d'hôte liés à la tâche
    1. 8.1.6.4.1 HostCallJobCallback( *callback* , *V* , *argumentsList* )
    2. 8.1.6.4.2 HostEnqueueFinalizationRegistryCleanupJob( *finalizationRegistry* )
    3. 8.1.6.4.3 HostEnqueuePromiseJob( *travail* , *domaine* )
    4. 8.1.6.4.4 HostMakeJobCallback( *appellable* )
  5. 8.1.6.5 Crochets d'hôte liés au module
    1. 8.1.6.5.1 HostGetImportMetaProperties( *moduleRecord* )
    2. 8.1.6.5.2 HostGetSupportedImportAssertions()
    3. 8.1.6.5.3 HostLoadImportedModule( *referrer* , *moduleRequest* , *loadState* , *payload* )
7. 8.1.7 Boucles d'événements
  1. 8.1.7.1 Définitions
  2. 8.1.7.2 Mise en file d'attente des tâches
  3. 8.1.7.3 Modèle de traitement
  4. 8.1.7.4 Sources de tâches génériques
  5. 8.1.7.5 Traitement de la boucle d'événements à partir d'autres spécifications
8. 8.1.8 Événements
  1. 8.1.8.1 Gestionnaires d'événements
  2. 8.1.8.2 Gestionnaires d'événements sur des éléments, *Document* des objets et *Window* des objets
    1. 8.1.8.2.1 Définitions IDL
  3. 8.1.8.3 Déclenchement d'événement
2. 8.2 Le *WindowOrWorkerGlobalScope* mixage
3. 8.3 Méthodes utilitaires Base64

4. 8.4 Insertion de balisage dynamique
  1. 8.4.1 Ouverture du flux d'entrée
  2. 8.4.2 Fermeture du flux d'entrée
  3. 8.4.3 `document.write()`
  4. 8.4.4 `document.writeln()`
5. 8.5 Analyse DOM
6. 8.6 Minuteries
7. 8.7 Mise en file d'attente des microtâches
8. 8.8 Invites de l'utilisateur
  1. 8.8.1 Boîtes de dialogue simples
  2. 8.8.2 Impression
9. 8.9 État et capacités du système
  1. 8.9.1 L' `Navigator` objet
    1. 8.9.1.1 Identification du client
    2. 8.9.1.2 Préférences de langue
    3. 8.9.1.3 État du navigateur
    4. 8.9.1.4 Gestionnaires de schémas personnalisés :  
la `registerProtocolHandler()` méthode
      1. 8.9.1.4.1 Sécurité et confidentialité
      2. 8.9.1.4.2 Automatisation de l'agent utilisateur
    5. 8.9.1.5 Cookies
    6. 8.9.1.6 Prise en charge de l'affichage PDF
10. 8.10 Images
11. 8.11 Images animées

## 9. 9 Communications

1. 9.1 L' `MessageEvent` interface
2. 9.2 Événements envoyés par le serveur
  1. 9.2.1 Présentation
  2. 9.2.2 L' `EventSource` interface
  3. 9.2.3 Modèle de traitement
  4. 9.2.4 L' `Last-Event-ID` en-tête ``
  5. 9.2.5 Analyser un flux d'événements
  6. 9.2.6 Interprétation d'un flux d'événements
  7. 9.2.7 Notes de création
  8. 9.2.8 Push sans connexion et autres fonctionnalités

- 9. 9.2.9 Collecte des ordures
  - 10.9.2.10 Conseils de mise en œuvre
- 3. 9.3 Messagerie inter-documents
  - 1. 9.3.1 Présentation
  - 2. 9.3.2 Sécurité
    - 1. 9.3.2.1 Auteurs
    - 2. 9.3.2.2 Agents utilisateurs
  - 3. 9.3.3 Publier des messages
- 4. 9.4 Messagerie du canal
  - 1. 9.4.1 Présentation
    - 1. 9.4.1.1 Exemples
    - 2. 9.4.1.2 Les ports comme base d'un modèle de capacité objet sur le Web
    - 3. 9.4.1.3 Ports comme base de l'abstraction des implémentations de service
  - 2. 9.4.2 Canaux de messages
  - 3. 9.4.3 Ports de messages
  - 4. 9.4.4 Diffusion vers de nombreux ports
  - 5. 9.4.5 Ports et collecte des ordures
- 5. 9.5 Diffusion vers d'autres contextes de navigation

## 10.10 travailleurs du Web

- 1. 10.1 Présentation
  - 1. 10.1.1 Portée
  - 2. 10.1.2 Exemples
    - 1. 10.1.2.1 Un travailleur de fond qui calcule les chiffres
    - 2. 10.1.2.2 Utiliser un module JavaScript en tant que worker
    - 3. 10.1.2.3 Présentation des travailleurs partagés
    - 4. 10.1.2.4 État partagé utilisant un travailleur partagé
    - 5. 10.1.2.5 Délégation
    - 6. 10.1.2.6 Fournir des bibliothèques
  - 3. 10.1.3 Tutoriels
    - 1. 10.1.3.1 Créer un worker dédié
    - 2. 10.1.3.2 Communiquer avec un intervenant dédié
    - 3. 10.1.3.3 Travailleurs partagés
- 2. 10.2 Infrastructures
  - 1. 10.2.1 Le périmètre mondial
    - 1. 10.2.1.1 L' `WorkerGlobalScope` interface commune

2. 10.2.1.2 Travailleurs dédiés  
et `DedicatedWorkerGlobalScope` interface
    3. 10.2.1.3 Travailleurs partagés  
et `SharedWorkerGlobalScope` interface
  2. 10.2.2 La boucle événementielle
  3. 10.2.3 La durée de vie du travailleur
  4. 10.2.4 Modèle de traitement
  5. 10.2.5 Erreurs de script d'exécution
  6. 10.2.6 Création de nœuds de calcul
    1. 10.2.6.1 Le `AbstractWorker` mélange
    2. 10.2.6.2 Paramètres de script pour les agents
    3. 10.2.6.3 Travailleurs dédiés et `Worker` interface
    4. 10.2.6.4 Travailleurs partagés et `SharedWorker` interface
  7. 10.2.7 Capacités matérielles simultanées
3. 10.3 API disponibles pour les travailleurs
  1. 10.3.1 Importation de scripts et de bibliothèques
  2. 10.3.2 L' `WorkerNavigator` interface
  3. 10.3.3 L' `WorkerLocation` interface

## 11.11 Worklets

1. 11.1 Présentation
  1. 11.1.1 Motivations
  2. 11.1.2 Code idempotence
  3. 11.1.3 Évaluation spéculative
2. 11.2 Exemples
  1. 11.2.1 Chargement des scripts
  2. 11.2.2 Enregistrement d'une classe et invocation de ses méthodes
3. 11.3 Infrastructures
  1. 11.3.1 Le périmètre mondial
    1. 11.3.1.1 Agents et boucles d'événements
    2. 11.3.1.2 Création et résiliation
    3. 11.3.1.3 Paramètres de script pour les worklets
  2. 11.3.2 La `Worklet` classe
  3. 11.3.3 Durée de vie du worklet

## 12.12 Stockage Web

1. 12.1 Présentation
2. 12.2 L'API
  1. 12.2.1 L' `Storage` interface
  2. 12.2.2 Le `sessionStorage` getter
  3. 12.2.3 Le `localStorage` getter
  4. 12.2.4 L' `StorageEvent` interface
3. 12.3 Confidentialité
  1. 12.3.1 Suivi des utilisateurs
  2. 12.3.2 Sensibilité des données
4. 12.4 Sécurité
  1. 12.4.1 Attaques d'usurpation de DNS
  2. 12.4.2 Attaques inter-répertoires
  3. 12.4.3 Risques de mise en œuvre

## 13.13 La syntaxe HTML

1. 13.1 Écrire des documents HTML
  1. 13.1.1 Le DOCTYPE
  2. 13.1.2 Éléments
    1. 13.1.2.1 Balises de début
    2. 13.1.2.2 Balises de fin
    3. 13.1.2.3 Attributs
    4. 13.1.2.4 Balises facultatives
    5. 13.1.2.5 Restrictions sur les modèles de contenu
    6. 13.1.2.6 Restrictions sur le contenu du texte brut et des éléments de texte brut échappables
  3. 13.1.3 Texte
    1. 13.1.3.1 Nouvelles lignes
  4. 13.1.4 Références de caractères
  5. 13.1.5 Rubriques CDATA
  6. 13.1.6 Commentaires
2. 13.2 Analyser des documents HTML
  1. 13.2.1 Présentation du modèle d'analyse syntaxique
  2. 13.2.2 Erreurs d'analyse
  3. 13.2.3 Le flux d'octets d'entrée
    1. 13.2.3.1 Analyse avec un codage de caractères connu
    2. 13.2.3.2 Détermination du codage des caractères
    3. 13.2.3.3 Encodages de caractères

4. 13.2.3.4 Modification de l'encodage lors de l'analyse
  5. 13.2.3.5 Prétraitement du flux d'entrée
4. 13.2.4 État d'analyse
  1. 13.2.4.1 Le mode d'insertion
  2. 13.2.4.2 L'empilement des éléments ouverts
  3. 13.2.4.3 La liste des éléments de formatage actifs
  4. 13.2.4.4 Les pointeurs d'éléments
  5. 13.2.4.5 Autres drapeaux d'état d'analyse
5. 13.2.5 Tokénisation
  1. 13.2.5.1 État des données
  2. 13.2.5.2 État RCDATA
  3. 13.2.5.3 Etat TEXTE BRUT
  4. 13.2.5.4 État des données de script
  5. 13.2.5.5 Etat TEXTE CLAIR
  6. 13.2.5.6 Etat ouvert de l'étiquette
  7. 13.2.5.7 Etat ouvert d'étiquette de fin
  8. 13.2.5.8 État du nom de variable
  9. 13.2.5.9 État de signe inférieur à RCDATA
  10. 13.2.5.10 État ouvert de l'étiquette de fin RCDATA
  11. 13.2.5.11 État du nom de balise de fin RCDATA
  12. 13.2.5.12 RAWTEXT état de signe inférieur à
  13. 13.2.5.13 État ouvert de la balise de fin RAWTEXT
  14. 13.2.5.14 État du nom de la balise de fin RAWTEXT
  15. 13.2.5.15 Etat des données de script inférieur à signe
  16. 13.2.5.16 État ouvert d'étiquette de fin de données de script
  17. 13.2.5.17 État du nom de balise de fin de données de script
  18. 13.2.5.18 Etat de début d'échappement des données de script
  19. 13.2.5.19 État du tiret de début d'échappement des données de script
  20. 13.2.5.20 État d'échappement des données de script
  21. 13.2.5.21 État du tiret échappé des données de script
  22. 13.2.5.22 Données de script échappées tiret état tiret
  23. 13.2.5.23 Les données de script ont échappé à l'état inférieur à signe
  24. 13.2.5.24 Etat ouvert de balise de fin échappée de données de script
  25. 13.2.5.25 Etat du nom de la balise de fin échappée des données de script
  26. 13.2.5.26 Etat de début d'échappement double des données de script

27. 13.2.5.27 État d'échappement double des données de script
28. 13.2.5.28 État du tiret à double échappement des données de script
29. 13.2.5.29 Données de script tiret à double échappement état tiret
30. 13.2.5.30 Les données de script ont échappé à deux fois à l'état de signe inférieur à
31. 13.2.5.31 État final d'échappement double des données de script
32. 13.2.5.32 Avant l'état du nom d'attribut
33. 13.2.5.33 État du nom d'attribut
34. 13.2.5.34 Après l'état du nom d'attribut
35. 13.2.5.35 Avant l'état de la valeur d'attribut
36. 13.2.5.36 État de la valeur d'attribut (entre guillemets)
37. 13.2.5.37 État de la valeur d'attribut (entre guillemets simples)
38. 13.2.5.38 État de valeur d'attribut (sans guillemets)
39. 13.2.5.39 Après l'état de valeur d'attribut (entre guillemets)
40. 13.2.5.40 État de la balise de début à fermeture automatique
41. 13.2.5.41 État de commentaire erroné
42. 13.2.5.42 État ouvert de la déclaration de balisage
43. 13.2.5.43 État de début de commentaire
44. 13.2.5.44 État du tiret de début de commentaire
45. 13.2.5.45 État des commentaires
46. 13.2.5.46 Commentaire état inférieur à signe
47. 13.2.5.47 Commentaire état de bang signe inférieur à
48. 13.2.5.48 Commentaire inférieur à signe bang état tiret
49. 13.2.5.49 Commentaire signe inférieur à bang tiret tiret état
50. 13.2.5.50 État du tiret de fin de commentaire
51. 13.2.5.51 État final du commentaire
52. 13.2.5.52 État du coup de fin de commentaire
53. 13.2.5.53 État DOCTYPE
54. 13.2.5.54 Avant l'état du nom DOCTYPE
55. 13.2.5.55 État du nom de DOCTYPE
56. 13.2.5.56 Après l'état du nom DOCTYPE
57. 13.2.5.57 Après l'état du mot clé public DOCTYPE
58. 13.2.5.58 Avant l'état de l'identificateur public DOCTYPE
59. 13.2.5.59 État de l'identificateur public DOCTYPE (entre guillemets)
60. 13.2.5.60 État de l'identificateur public DOCTYPE (entre guillemets simples)



61. 13.2.5.61 Après l'état de l'identificateur public DOCTYPE
62. 13.2.5.62 Entre l'état des identifiants public et système de DOCTYPE
63. 13.2.5.63 Après l'état du mot clé système DOCTYPE
64. 13.2.5.64 Avant l'état de l'identificateur de système DOCTYPE
65. 13.2.5.65 État de l'identificateur de système DOCTYPE (entre guillemets)
66. 13.2.5.66 État de l'identificateur système DOCTYPE (entre guillemets simples)
67. 13.2.5.67 Après l'état de l'identificateur de système DOCTYPE
68. 13.2.5.68 État DOCTYPE erroné
69. 13.2.5.69 État de la section CDATA
70. 13.2.5.70 État du support de section CDATA
71. 13.2.5.71 État final de la section CDATA
72. 13.2.5.72 Etat de référence des caractères
73. 13.2.5.73 Etat de référence du caractère nommé
74. 13.2.5.74 Esperluette ambiguë
75. 13.2.5.75 Etat de référence des caractères numériques
76. 13.2.5.76 Etat de début de référence de caractère hexadécimal
77. 13.2.5.77 Etat de début de référence de caractère décimal
78. 13.2.5.78 Etat de référence des caractères hexadécimaux
79. 13.2.5.79 Etat de référence du caractère décimal
80. 13.2.5.80 Etat final de référence de caractère numérique
6. 13.2.6 Construction d'arbres
  1. 13.2.6.1 Création et insertion de nœuds
  2. 13.2.6.2 Analyser des éléments qui ne contiennent que du texte
  3. 13.2.6.3 Éléments de fermeture qui ont des balises de fin implicites
  4. 13.2.6.4 Les règles d'analyse des jetons dans le contenu HTML
    1. 13.2.6.4.1 Le mode d'insertion "initial"
    2. 13.2.6.4.2 Le mode d'insertion "avant html"
    3. 13.2.6.4.3 Le mode d'insertion "avant tête"
    4. 13.2.6.4.4 Le mode d'insertion "en tête"
    5. 13.2.6.4.5 Le mode d'insertion "in head noscript"
    6. 13.2.6.4.6 Le mode d'insertion "après tête"
    7. 13.2.6.4.7 Le mode d'insertion "dans le corps"
    8. 13.2.6.4.8 Le mode d'insertion "texte"
    9. 13.2.6.4.9 Le mode d'insertion "dans le tableau"
    10. 13.2.6.4.10 Le mode d'insertion "dans le texte du tableau"

11. 13.2.6.4.11 Le mode d'insertion "en légende"
12. 13.2.6.4.12 Le mode d'insertion "dans le groupe de colonnes"
13. 13.2.6.4.13 Le mode d'insertion "dans le corps du tableau"
14. 13.2.6.4.14 Le mode d'insertion "en ligne"
15. 13.2.6.4.15 Le mode d'insertion "dans cellule"
16. 13.2.6.4.16 Le mode d'insertion "en sélection"
17. 13.2.6.4.17 Le mode d'insertion "dans select in table"
18. 13.2.6.4.18 Le mode d'insertion "dans le modèle"
19. 13.2.6.4.19 Le mode d'insertion "après le corps"
20. 13.2.6.4.20 Le mode d'insertion "in frameset"
21. 13.2.6.4.21 Le mode d'insertion "après frameset"
22. 13.2.6.4.22 Le mode d'insertion "après après corps"
23. 13.2.6.4.23 Le mode d'insertion "après après frameset"
5. 13.2.6.5 Les règles d'analyse des jetons dans le contenu étranger
7. 13.2.7 La fin
8. 13.2.8 Analyse HTML spéculative
9. 13.2.9 Contrainte d'un DOM HTML dans un ensemble d'informations
10. 13.2.10 Une introduction à la gestion des erreurs et des cas étranges dans l'analyseur
  1. 13.2.10.1 Balises mal imbriquées : `<b><i></b></i>`
  2. 13.2.10.2 Balises mal imbriquées : `<b><p></b></p>`
  3. 13.2.10.3 Balisage inattendu dans les tableaux
  4. 13.2.10.4 Scripts qui modifient la page lors de son analyse
  5. 13.2.10.5 L'exécution de scripts qui se déplacent sur plusieurs documents
  6. 13.2.10.6 Éléments de formatage non fermés
3. 13.3 Sérialisation de fragments HTML
4. 13.4 Analyser des fragments HTML
5. 13.5 Références de caractères nommés

## 14.14 La syntaxe XML

1. 14.1 Écrire des documents dans la syntaxe XML
2. 14.2 Analyser des documents XML
3. 14.3 Sérialisation des fragments XML

#### 4. 14.4 Analyser des fragments XML

### 15.15 Rendu

1. 15.1 Présentation
2. 15.2 La feuille de style de l'agent utilisateur CSS et les conseils de présentation
3. 15.3 Éléments non remplacés
  1. 15.3.1 Éléments cachés
  2. 15.3.2 La page
  3. 15.3.3 Contenu du flux
  4. 15.3.4 Phrase contenu
  5. 15.3.5 Texte bidirectionnel
  6. 15.3.6 Sections et titres
  7. 15.3.7 Listes
  8. 15.3.8 Tableaux
  9. 15.3.9 bizarreries d'effondrement de marge
  10. 15.3.10 Champs de formulaire
  11. 15.3.11 L' `hr` élément
  12. 15.3.12 Les éléments `fieldset` et `legend`
4. 15.4 Éléments remplacés
  1. 15.4.1 Contenu intégré
  2. 15.4.2 Images
  3. 15.4.3 Attributs pour le contenu intégré et les images
  4. 15.4.4 Images cliquables
5. 15.5 Gadgets
  1. 15.5.1 Apparence native
  2. 15.5.2 Disposition des boutons
  3. 15.5.3 L' `button` élément
  4. 15.5.4 Les éléments `details` et `summary`
  5. 15.5.5 L' `input` élément en tant que widget de saisie de texte
  6. 15.5.6 L' `input` élément en tant que widgets spécifiques à un domaine
  7. 15.5.7 L' `input` élément comme contrôle de distance
  8. 15.5.8 L' `input` élément comme puits de couleur
  9. 15.5.9 L' `input` élément en tant que widgets de case à cocher et de bouton radio

10. 15.5.10 L' `input` élément en tant que contrôle de téléchargement de fichier
11. 15.5.11 L' `input` élément sous forme de bouton
12. 15.5.12 L' `marquee` élément
13. 15.5.13 L' `meter` élément
14. 15.5.14 L' `progress` élément
15. 15.5.15 L' `select` élément
16. 15.5.16 L' `textarea` élément
6. 15.6 Cadres et jeux de cadres
7. 15.7 Médias interactifs
  1. 15.7.1 Liens, formulaires et navigation
  2. 15.7.2 L' `title` attribut
  3. 15.7.3 Modification des hôtes
  4. 15.7.4 Texte rendu dans les interfaces utilisateur natives
8. 15.8 Supports d'impression
9. 15.9 Documents XML sans style

## 16.16 fonctionnalités obsolètes

1. 16.1 Caractéristiques obsolètes mais conformes
  1. 16.1.1 Avertissements pour les fonctionnalités obsolètes mais conformes
2. 16.2 Éléments non conformes
3. 16.3 Exigences pour les implémentations
  1. 16.3.1 L' `marquee` élément
  2. 16.3.2 Cadres
  3. 16.3.3 Autres éléments, attributs et API

## 17.17 Considérations relatives à l'IANA

1. 17.1 `text/html`
2. 17.2 `multipart/x-mixed-replace`
3. 17.3 `application/xhtml+xml`
4. 17.4 `text/ping`
5. 17.5 `application/microdata+json`
6. 17.6 `text/event-stream`

7. 17.7 [web+](#) préfixe de schéma

18. [Indice](#)

1. [Éléments](#)
2. [Catégories de contenu d'élément](#)
3. [Les attributs](#)
4. [Interfaces d'éléments](#)
5. [Toutes les interfaces](#)
6. [Événements](#)
7. [En-têtes HTTP](#)
8. [types MIME](#)

19. [Les références](#)

20. [Remerciements](#)

21. [Droits de propriété intellectuelle](#)