eBook Gratuit

APPRENEZ CSS

eBook gratuit non affilié créé à partir des **contributeurs de Stack Overflow.**

#css

Table des matières

À propos

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: css

It is an unofficial and free CSS ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official CSS.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapitre 1: Démarrer avec CSS

Remarques

Les styles peuvent être créés de plusieurs manières, ce qui permet divers degrés de réutilisation et de portée lorsqu'ils sont spécifiés dans un document HTML source. *Les* feuilles de style *externes* peuvent être réutilisées dans les documents HTML. *Les* feuilles de style *incorporées* s'appliquent à l'ensemble du document dans lequel elles sont spécifiées. *Les* styles en *ligne* s'appliquent uniquement à l'élément HTML individuel sur lequel ils sont spécifiés.

Versions

Version	Date de sortie
1	1996-12-17
2	1998-05-12
3	2015-10-13

Examples

Feuille de style externe

Une feuille de style CSS externe peut être appliquée à un nombre quelconque de documents HTML en plaçant un élément `<link>` dans chaque document HTML.

L'attribut `rel` de la `<link>` doit être défini sur `"stylesheet"` et l'attribut `href` sur le chemin relatif ou absolu de la feuille de style. Bien que l'utilisation de chemins d'URL relatifs soit généralement considérée comme une bonne pratique, des chemins absolus peuvent également être utilisés. En HTML5, l'attribut `type` peut être omis .

Il est recommandé de placer la balise `<link>` dans la balise `<head>` du fichier HTML afin que les styles soient chargés avant les éléments de style. Sinon, les utilisateurs verront un flash de contenu non stylé .

Exemple

hello-world.html

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8" />
```

```
        <link rel="stylesheet" type="text/css" href="style.css">
    </head>
    <body>
        <h1>Hello world!</h1>
        <p>I ♥ CSS</p>
  </body>
</html>
```

style.css

```
h1 {
    color: green;
    text-decoration: underline;
}
p {
    font-size: 25px;
    font-family: 'Trebuchet MS', sans-serif;
}
```

Assurez-vous d'inclure le chemin d'accès correct à votre fichier CSS dans le href. Si le fichier CSS se trouve dans le même dossier que votre fichier HTML, aucun chemin n'est requis (comme dans l'exemple ci-dessus), mais s'il est enregistré dans un dossier, spécifiez-le comme ceci:
`href="foldername/style.css"` .

```
<link rel="stylesheet" type="text/css" href="foldername/style.css">
```

Les feuilles de style externes sont considérées comme le meilleur moyen de gérer votre CSS. Il y a une raison très simple à cela: lorsque vous gérez un site de 100 pages, par exemple, toutes contrôlées par une seule feuille de style et que vous souhaitez modifier les couleurs de votre lien du bleu au vert, il est beaucoup plus facile de faire la modification. dans votre fichier CSS et laissez les changements "en cascade" dans toutes les 100 pages que dans 100 pages distinctes et faites le même changement 100 fois. Encore une fois, si vous souhaitez modifier complètement l'aspect de votre site Web, il vous suffit de mettre à jour ce fichier.

Vous pouvez charger autant de fichiers CSS dans votre page HTML que nécessaire.

```
<link rel="stylesheet" type="text/css" href="main.css">
<link rel="stylesheet" type="text/css" href="override.css">
```

Les règles CSS sont appliquées avec des règles de base, et l'ordre compte. Par exemple, si vous avez un fichier main.css contenant du code:

```
p.green { color: #00FF00; }
```

Tous vos paragraphes avec la classe 'green' seront écrits en vert clair, mais vous pouvez remplacer cela par un autre fichier .css en l'incluant juste *après* main.css. Vous pouvez avoir override.css avec le code suivant, suivi de main.css, par exemple:

```
p.green { color: #006600; }
```

Maintenant, tous vos paragraphes avec la classe «verte» seront écrits en vert plus foncé plutôt qu'en vert clair.

D'autres principes s'appliquent, tels que la règle «! Important», la spécificité et l'héritage.

Lorsque quelqu'un visite votre site Web pour la première fois, son navigateur télécharge le code HTML de la page en cours et le fichier CSS lié. Ensuite, lorsqu'ils naviguent vers une autre page, leur navigateur doit uniquement télécharger le code HTML de cette page. le fichier CSS est mis en cache, il n'est donc pas nécessaire de le télécharger à nouveau. Comme les navigateurs mettent en cache la feuille de style externe, vos pages se chargent plus rapidement.

Styles internes

Le `<style></style>` CSS inséré dans les balises `<style></style>` dans un document HTML fonctionne comme une feuille de style externe, sauf qu'il se trouve dans le document HTML qu'il filtre plutôt que dans un fichier distinct. Il ne peut donc être appliqué qu'au document dans lequel il se trouve. vies. Notez que cet élément *doit* être dans l'élément `<head>` pour la validation HTML (bien qu'il fonctionnera dans tous les navigateurs actuels s'il est placé dans le `body`).

```
<head>
    <style>
        h1 {
            color: green;
            text-decoration: underline;
        }
        p {
            font-size: 25px;
            font-family: 'Trebuchet MS', sans-serif;
        }
    </style>
</head>
<body>
    <h1>Hello world!</h1>
    <p>I ♥ CSS</p>
</body>
```

Styles en ligne

Utilisez des styles en ligne pour appliquer un style à un élément spécifique. Notez que ce n'est ***pas*** optimal. Placer les règles de style dans une `<style>` ou un fichier CSS externe est encouragé afin de maintenir une distinction entre le contenu et la présentation.

Les styles en ligne remplacent les CSS dans une `<style>` ou une feuille de style externe. Bien que cela puisse être utile dans certaines circonstances, ce fait réduit le plus souvent la maintenabilité d'un projet.

Les styles de l'exemple suivant s'appliquent directement aux éléments auxquels ils sont attachés.

```
<h1 style="color: green; text-decoration: underline;">Hello world!</h1>
<p style="font-size: 25px; font-family: 'Trebuchet MS';">I ♥ CSS</p>
```

Les styles en ligne sont généralement le moyen le plus sûr d'assurer la compatibilité du rendu entre différents clients, programmes et périphériques de messagerie, mais leur écriture peut être longue et difficile à gérer.

CSS @import rule (une règle CSS)

La règle @import CSS est utilisée pour importer des règles de style provenant d'autres feuilles de style. Ces règles doivent précéder tous les autres types de règles, à l'exception des règles @charset; Comme il ne s'agit pas d'une instruction imbriquée, @import ne peut pas être utilisé dans les règles de groupe conditionnelles. `@import` .

Comment utiliser @import

Vous pouvez utiliser la règle @import des manières suivantes:

A. avec étiquette de style interne

```
<style>
 @import url('/css/styles.css');
</style>
```

B. avec feuille de style externe

La ligne suivante importe un fichier CSS nommé `additional-styles.css` dans le répertoire racine dans le fichier CSS dans lequel il apparaît:

```
@import '/additional-styles.css';
```

L'importation de CSS externes est également possible. Les fichiers de polices sont un cas d'utilisation courant.

```
@import 'https://fonts.googleapis.com/css?family=Lato';
```

Un second argument facultatif à la règle `@import` est une liste de requêtes de média:

```
@import '/print-styles.css' print;
@import url('landscape.css') screen and (orientation:landscape);
```

Changer de CSS avec JavaScript

JavaScript pur

Il est possible d'ajouter, de supprimer ou de modifier les valeurs de propriété CSS avec JavaScript via la propriété de `style` un élément.

```
var el = document.getElementById("element");
el.style.opacity = 0.5;
```

```
el.style.fontFamily = 'sans-serif';
```

Notez que les propriétés de style sont nommées dans le style de casse camel inférieur. Dans l'exemple, vous voyez que la propriété `font-family` propriété css devient `fontFamily` en javascript.

Au lieu de travailler directement sur des éléments, vous pouvez créer un élément `<style>` ou `<link>` dans JavaScript et l'ajouter à `<body>` ou `<head>` du document HTML.

jQuery

La modification des propriétés CSS avec jQuery est encore plus simple.

```
$('#element').css('margin', '5px');
```

Si vous devez modifier plusieurs règles de style:

```
$('#element').css({
    margin: "5px",
    padding: "10px",
    color: "black"
});
```

jQuery inclut deux façons de modifier les règles CSS contenant des tirets (c. `font-size`). Vous pouvez les mettre entre guillemets ou avec le nom de la règle de style.

```
$('.example-class').css({
    "background-color": "blue",
    fontSize: "10px"
});
```

Voir également

- Documentation JavaScript - Lecture et modification du style CSS .
- Documentation jQuery - Manipulation CSS

Listes de styles avec CSS

Il existe trois propriétés différentes pour les éléments de `list-style-type` : `list-style-type` , `list-style-image` et `list-style-position` , qui doivent être déclarés dans cet ordre. Les valeurs par défaut sont respectivement disque, extérieur et aucun. Chaque propriété peut être déclarée séparément ou en utilisant la propriété raccourcie de `list-style` .

`list-style-type` définit la forme ou le type de point utilisé pour chaque élément de liste.

Certaines des valeurs acceptables pour `list-style-type` :

- disque

- cercle
- carré
- décimal
- bas-roman
- haut-roman
- aucun

(Pour une liste exhaustive, voir le wiki de spécification du W3C)

Pour utiliser des puces carrées pour chaque élément de liste, par exemple, vous utiliseriez la paire propriété-valeur suivante:

```
li {
    list-style-type: square;
}
```

La propriété **`list-style-image`** détermine si l'icône de l'élément de liste est définie avec une image et accepte une valeur `none` ou une URL qui pointe vers une image.

```
li {
    list-style-image: url(images/bullet.png);
}
```

La propriété **`list-style-position`** définit où placer le marqueur d'élément de liste et accepte l'une des deux valeurs suivantes: "inside" ou "outside".

```
li {
  list-style-position: inside;
}
```

Lire Démarrer avec CSS en ligne: https://riptutorial.com/fr/css/topic/293/demarrer-avec-css

Chapitre 2: Ajustement d'objet et placement

Remarques

Les propriétés `object-fit` et `object-position` ne sont pas prises en charge par Internet Explorer.

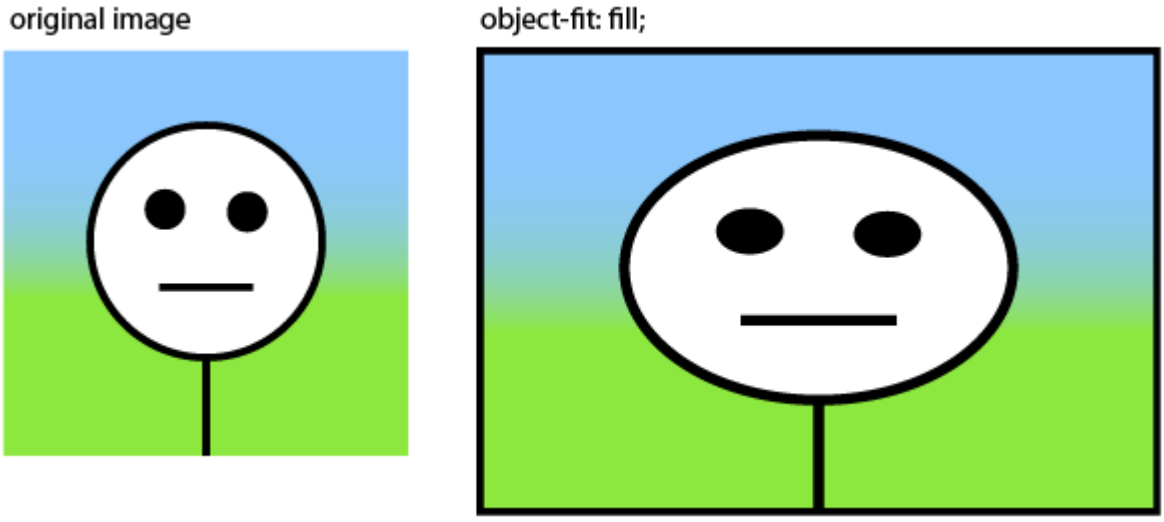
Examples

ajustement d'objet

La propriété **object-fit** définira comment un élément s'intégrera dans une boîte avec une hauteur et une largeur établies. Généralement appliqué à une image ou à une vidéo, Object-fit accepte les cinq valeurs suivantes:

REMPLIR

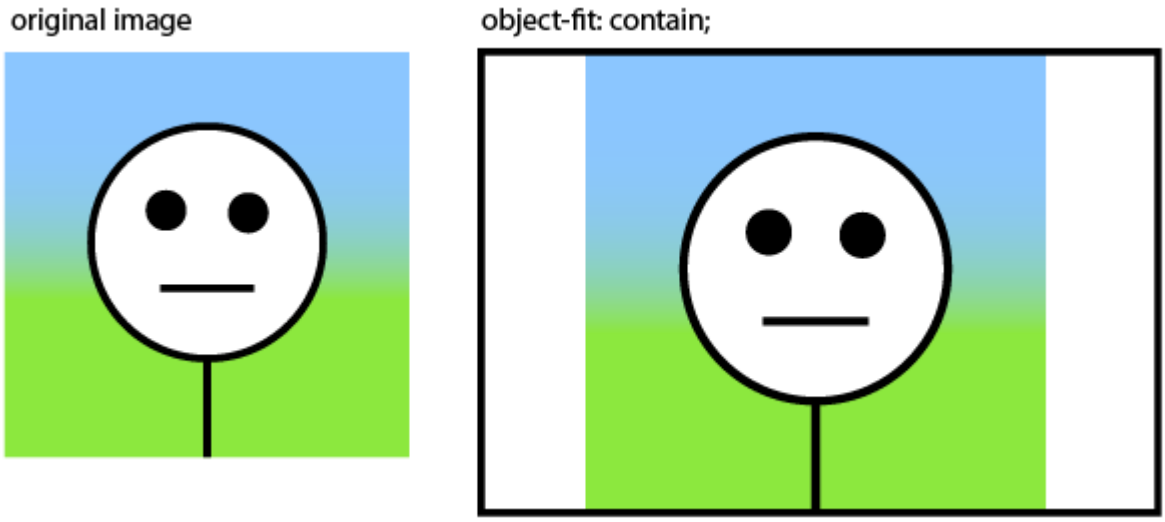
```
object-fit:fill;
```



Fill étend l'image pour l'adapter à la zone de contenu, indépendamment du format d'origine de l'image.

CONTENIR

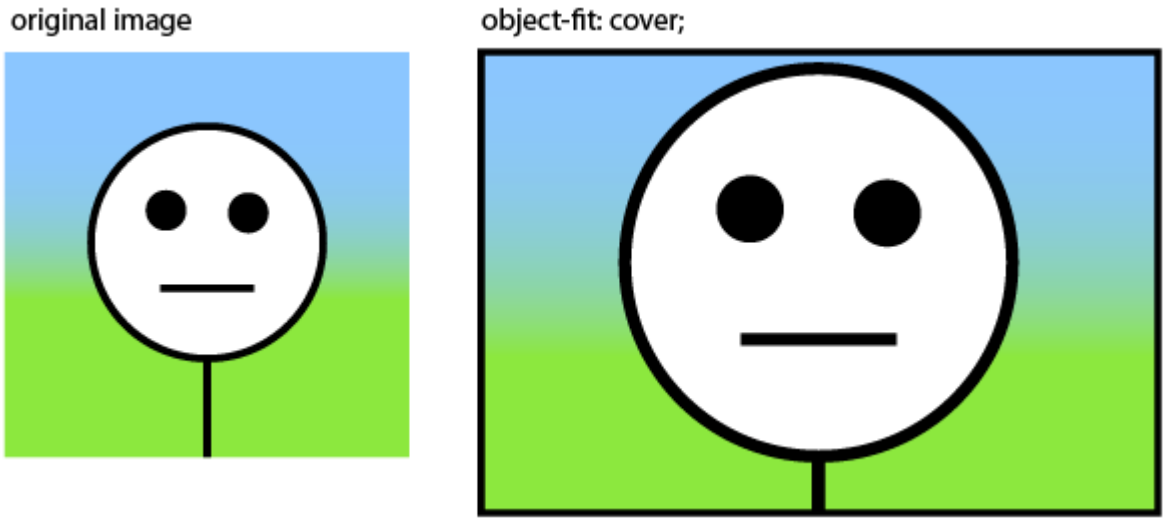
```
object-fit:contain;
```



Contient adapte l'image à la hauteur ou à la largeur de la boîte tout en conservant les proportions de l'image.

COUVERTURE

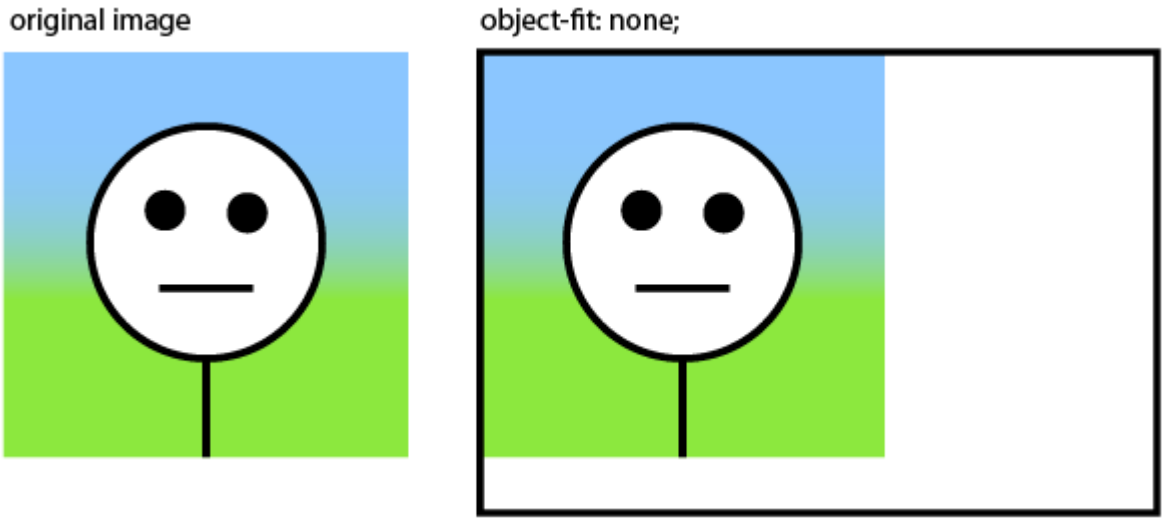
```
object-fit:cover;
```



La couverture remplit la boîte entière avec l'image. Le format de l'image est préservé, mais l'image est rognée aux dimensions de la boîte.

AUCUN

```
object-fit:none;
```

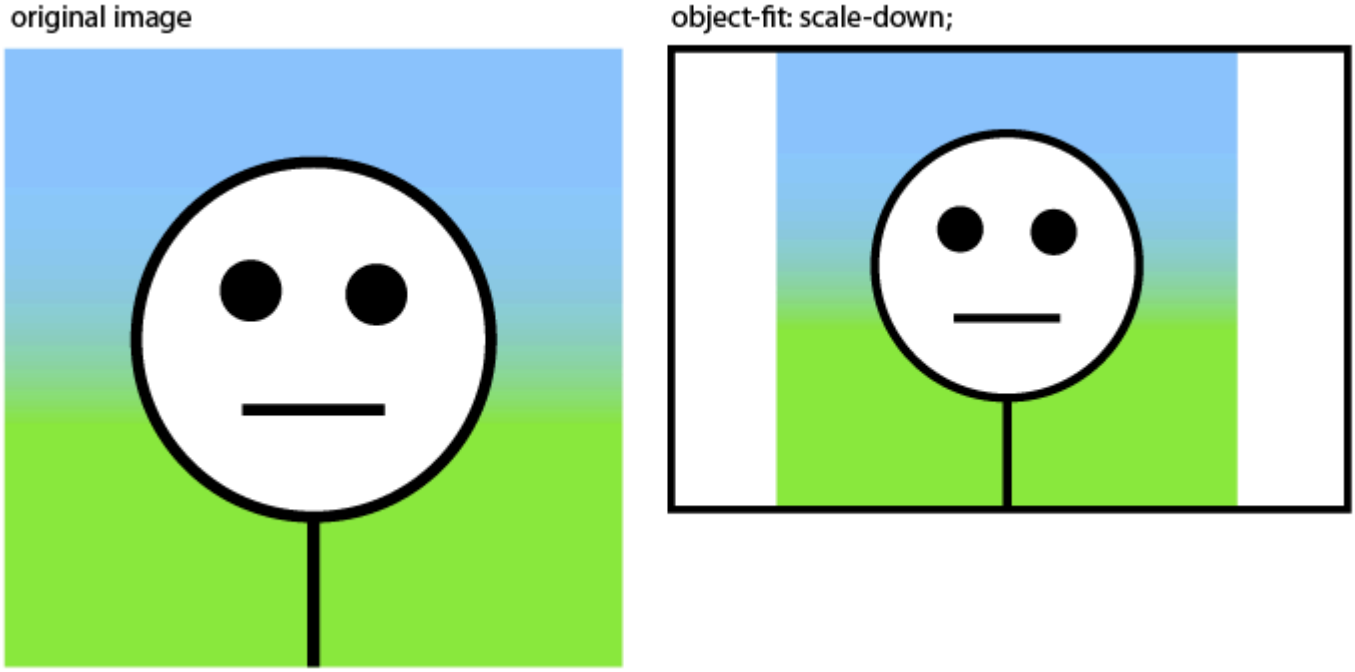


Aucun ignore la taille de la boîte et n'est pas redimensionné.

SCALE-DOWN

```
object-fit:scale-down;
```

Scale-down redimensionne l'objet comme `none` ou `contain` . Il affiche toutes les options entraînant une taille d'image plus petite.



Lire Ajustement d'objet et placement en ligne: https://riptutorial.com/fr/css/topic/5520/ajustement-d-objet-et-placement

Chapitre 3: Arrière-plans

Introduction

Avec CSS, vous pouvez définir les couleurs, les dégradés et les images comme arrière-plan d'un élément.

Il est possible de spécifier différentes combinaisons d'images, de couleurs et de dégradés, et d'ajuster la taille, le positionnement et la répétition (entre autres).

Syntaxe

- couleur de fond: couleur | transparent | initiale | hériter;
- background-image: url | aucun initiale | hériter;
- background-position: valeur;
- taille d'arrière-plan: <bg-size> [<bg-size>]
- <bg-size>: auto | longueur | couverture | contenir | initiale | hériter;
- répétition d'arrière-plan: répéter | repeat-x | répétition-y | pas de répétition | initiale | hériter;
- background-origine: padding-box | border-box | box de contenu | initiale | hériter;
- background-clip: border-box | boîte de rembourrage | box de contenu | initiale | hériter;
- pièce jointe: défilement | fixe | local | initiale | hériter;
- arrière-plan: bg-image position bg-image / taille bg répétition bg origine bg-clip fixation bg initiale | hériter;

Remarques

- Les dégradés CSS3 ne fonctionneront pas sur les versions d'Internet Explorer inférieures à 10.

Examples

Couleur de fond

La propriété `background-color` définit la couleur d'arrière-plan d'un élément à l'aide d'une valeur de couleur ou de mots-clés, tels que `transparent` , `inherit` ou `initial` .

- **transparent** , spécifie que la couleur d'arrière-plan doit être transparente. Ceci est la valeur par défaut.
- **inherit** , hérite de cette propriété de son élément parent.
- **initial** , définit cette propriété sur sa valeur par défaut.

Cela peut être appliqué à tous les éléments, et aux pseudo-éléments `::first-letter` / `::first-line` .

Les couleurs en CSS peuvent être spécifiées par différentes méthodes .

Les noms de couleur

CSS

```
div {
  background-color: red;  /* red */
}
```

HTML

```
<div>This will have a red background</div>
```

- L'exemple ci-dessus est l'une des nombreuses façons dont CSS doit représenter une seule couleur.

Codes de couleur hexadécimaux

Le code hexadécimal est utilisé pour désigner les composants RVB d'une couleur en notation hexadécimale de base 16. # ff0000, par exemple, est rouge vif, le composant rouge de la couleur est 256 bits (ff) et les parties verte et bleue correspondantes de la couleur sont 0 (00).

Si les deux valeurs de chacun des trois couplages RVB (R, G et B) sont identiques, le code de couleur peut être raccourci en trois caractères (le premier chiffre de chaque association). `#ff0000` peut être raccourci en `#f00` , et `#ffffff` peut être raccourci en `#fff` .

La notation hexadécimale est insensible à la casse.

```
body {
  background-color: #de1205; /* red */
}

.main {
  background-color: #00f; /* blue */
}
```

RGB / RGBa

Une autre façon de déclarer une couleur est d'utiliser RGB ou RGBa.

RVB signifie Rouge, Vert et Bleu, et nécessite trois valeurs distinctes entre 0 et 255, mises entre parenthèses, qui correspondent aux valeurs de couleur décimales pour respectivement rouge, vert et bleu.

RGBa vous permet d'ajouter un paramètre alpha supplémentaire entre 0.0 et 1.0 pour définir

l'opacité.

```
header {
  background-color: rgb(0, 0, 0); /* black */
}

footer {
  background-color: rgba(0, 0, 0, 0.5); /* black with 50% opacity */
}
```

HSL / HSLa

Une autre façon de déclarer une couleur est d'utiliser HSL ou HSLa et est similaire à RGB et RGBa.

HSL est synonyme de teinte, de saturation et de légèreté et est souvent appelé HLS:

- La teinte est un degré sur la roue chromatique (de 0 à 360).
- La saturation est un pourcentage compris entre 0% et 100%.
- La légèreté est également un pourcentage compris entre 0% et 100%.

HSLa vous permet d'ajouter un paramètre alpha supplémentaire entre 0.0 et 1.0 pour définir l'opacité.

```
li a {
  background-color: hsl(120, 100%, 50%); /* green */
}

#p1 {
  background-color: hsla(120, 100%, 50%, .3); /* green with 30% opacity */
}
```

Interaction avec image de fond

Les instructions suivantes sont toutes équivalentes:

```
body {
  background: red;
  background-image: url(partiallytransparentimage.png);
}

body {
  background-color: red;
  background-image: url(partiallytransparentimage.png);
}

body {
  background-image: url(partiallytransparentimage.png);
  background-color: red;
}
```

```
body {
  background: red url(partiallytransparentimage.png);
}
```

Ils mèneront tous à la couleur rouge affichée sous l'image, où les parties de l'image sont transparentes, ou l'image ne s'affiche pas (peut-être à la suite d' `background-repeat`).

Notez que ce qui suit n'est pas équivalent:

```
body {
  background-image: url(partiallytransparentimage.png);
  background: red;
}
```

Ici, la valeur d' `background` - `background` remplace votre `background-image` .

Pour plus d'informations sur la propriété d' `background` - `background` , voir Abrégé d'arrière - plan

Image de fond

La propriété `background-image` est utilisée pour spécifier une image d'arrière-plan à appliquer à tous les éléments correspondants. Par défaut, cette image est en mosaïque pour couvrir l'élément entier, à l'exclusion de la marge.

```
.myClass {
  background-image: url('/path/to/image.jpg');
}
```

Pour utiliser plusieurs images en tant qu'image d' `background-image` , définissez `url()`

```
.myClass {
  background-image: url('/path/to/image.jpg'),
                    url('/path/to/image2.jpg');
}
```

Les images s'empileront selon leur ordre avec la première image déclarée sur les autres et ainsi de suite.

Valeur	Résultat
`url('/path/to/image.jpg')`	Spécifiez le (s) chemin (s) de l'image d'arrière-plan ou une ressource d'image spécifiée avec le schéma URI de données (les apostrophes peuvent être omises), séparez les multiples par une virgule
`none`	Pas d'image de fond
`initial`	Valeur par défaut
`inherit`	Hériter la valeur du parent

Plus CSS pour Image d'arrière-plan

Ces attributs suivants sont très utiles et presque essentiels.

```
background-size:     xpx ypx | x% y%;
background-repeat:   no-repeat | repeat | repeat-x | repeat-y;
background-position: left offset (px/%) right offset (px/%) | center center | left top | right
bottom;
```

Dégradés de fond

Les dégradés sont de nouveaux types d'image, ajoutés dans CSS3. En tant qu'image, les dégradés sont définis avec la propriété `background-image` ou le raccourci d' `background` - `background` .

Il existe deux types de fonctions de gradient, linéaires et radiales. Chaque type a une variante non répétitive et une variante répétitive:

- `linear-gradient()`
- `repeating-linear-gradient()`
- `radial-gradient()`
- `repeating-radial-gradient()`

gradient linéaire()

Un `linear-gradient` a la syntaxe suivante

```
background: linear-gradient( <direction>?, <color-stop-1>, <color-stop-2>, ...);
```

Valeur	Sens
`<direction>`	Pourrait être un argument comme `to top` , `to bottom` , `to right` ou `to left` ; ou un angle comme `0deg` , `90deg` .... L'angle commence de haut en bas et tourne dans le sens des aiguilles d'une montre. Peut être spécifié en deg , grad , rad ou turn . Si omis, le dégradé s'écoule de haut en bas
`<color-stop-list>`	Liste de couleurs, éventuellement suivie chacune d'un pourcentage ou d'une longueur pour l'afficher à. Par exemple, `yellow 10%` , `rgba(0,0,0,.5) 40px` , `#fff 100%` ...

Par exemple, cela crée un dégradé linéaire qui part de la droite et passe du rouge au bleu

```
.linear-gradient {
  background: linear-gradient(to left, red, blue); /* you can also use 270deg */
}
```

Vous pouvez créer un dégradé en `diagonal` en déclarant une position de départ horizontale et verticale.

```
.diagonal-linear-gradient {
  background: linear-gradient(to left top, red, yellow 10%);
}
```

Il est possible de spécifier n'importe quel nombre d'arrêts de couleur dans un dégradé en les séparant par des virgules. Les exemples suivants créeront un dégradé avec 8 arrêts de couleur

```
.linear-gradient-rainbow {
  background: linear-gradient(to left, red, orange, yellow, green, blue, indigo, violet)
}
```

gradient radial ()

```
.radial-gradient-simple {
  background: radial-gradient(red, blue);
}

.radial-gradient {
  background: radial-gradient(circle farthest-corner at top left, red, blue);
}
```

Valeur	Sens
`circle`	Forme du dégradé. Les valeurs sont `circle` ou `ellipse` , la valeur par défaut est `ellipse` .
`farthest-corner`	Mots-clés décrivant la taille de la forme finale. Les valeurs sont les `closest-side` , les `farthest-side` , les `farthest-side closest-corner` , les `farthest-corner`
`top left`	Définit la position du centre du dégradé, de la même manière que la `background-position` en `background-position` .

Gradients répétés

Les fonctions de dégradations répétées prennent les mêmes arguments que les exemples ci-dessus, mais placent le dégradé dans l'arrière-plan de l'élément.

```
.bullseye {
  background: repeating-radial-gradient(red, red 10%, white 10%, white 20%);
}
.warning {
  background: repeating-linear-gradient(-45deg, yellow, yellow 10%, black 10%, black 20% );
}
```

Valeur	Sens
`-45deg`	Unité d'angle . L'angle commence par le haut et tourne dans le sens des aiguilles d'une montre. Peut être spécifié en deg , grad , rad ou turn .
`to left`	Direction du dégradé, par défaut `to bottom` . Syntaxe: `to [y-axis(top OR bottom)] [x-axis(left OR right)]` c'est- `to top right` dire `to top right`
`yellow 10%`	Couleur, éventuellement suivie d'un pourcentage ou d'une longueur pour l'afficher à. Répété deux fois ou plus.

Notez que les codes de couleur HEX, RGB, RGBa, HSL et HSLa peuvent être utilisés à la place des noms de couleur. Les noms de couleurs ont été utilisés à des fins d'illustration. Notez également que la syntaxe à gradient radial est beaucoup plus complexe que le gradient linéaire, et une version simplifiée est présentée ici. Pour une explication complète et les spécifications, consultez la documentation MDN

Sténographie de fond

La propriété `background` peut être utilisée pour définir une ou plusieurs propriétés liées à l'arrière-plan:

Valeur	La description	CSS Ver.
`background-image`	Image de fond à utiliser	1+
`background-color`	Couleur de fond à appliquer	1+
`background-position`	Position de l'image d'arrière-plan	1+
`background-size`	Taille de l'image de fond	3+
`background-repeat`	Comment répéter l'image de fond	1+
`background-origin`	Comment l'arrière-plan est positionné (ignoré lorsque l' `background-attachment` est `fixed`)	3+
`background-clip`	Comment l'arrière - plan est peint par rapport à la `content-box` , `border-box` , ou `padding-box`	3+
`background-attachment`	Comment se comporte l'image d'arrière-plan, qu'elle défile avec son bloc conteneur ou qu'elle ait une position fixe dans la fenêtre d'affichage	1+
`initial`	Définit la propriété sur valeur par défaut	3+
`inherit`	Hérite la valeur de propriété du parent	2+

L'ordre des valeurs n'a pas d'importance et chaque valeur est facultative

Syntaxe

La syntaxe de la déclaration abrégée d'arrière-plan est la suivante:

```
background: [<background-image>] [<background-color>]  [<background-position>]/[<background-size>] [<background-repeat>] [<background-origin>] [<background-clip>] [<background-attachment>] [<initial|inherit>];
```

Exemples

```
background: red;
```

Il suffit de définir une `background-color` d' `background-color` avec la valeur `red` .

```
background: border-box red;
```

Définir un `background-clip` d' `background-clip` sur border-box et une `background-color` d' `background-color` sur rouge.

```
background: no-repeat center url("somepng.jpg");
```

Définit une `background-repeat` d' `background-repeat` sur non-répétition, `background-origin` d' `background-origin` au centre et `background-image` d' `background-image` sur une image.

```
background: url('pattern.png') green;
```

Dans cet exemple, la `background-color` d' `background-color` de l'élément serait définie sur `green` avec `pattern.png` , s'il est disponible, sur la couleur, en répétant autant de fois que nécessaire pour remplir l'élément. Si `pattern.png` inclut une transparence, la couleur `green` sera visible derrière.

```
background: #000000 url("picture.png") top left / 600px auto no-repeat;
```

Dans cet exemple, nous avons un fond noir avec une image "picture.png" en haut, l'image ne se répète pas dans les deux axes et est positionnée dans le coin supérieur gauche. Le `/` après la position est de pouvoir inclure la taille de l'image de fond qui, dans ce cas, est définie comme largeur `600px` et auto pour la hauteur. Cet exemple pourrait bien fonctionner avec une image de fonction qui peut devenir une couleur unie.

> **REMARQUE: L'** utilisation de la propriété d'arrière-plan abrégée réinitialise toutes les valeurs de propriété d'arrière-plan définies précédemment, même si aucune valeur n'est donnée. Si vous souhaitez uniquement modifier une valeur de propriété d'arrière-plan définie précédemment, utilisez plutôt une propriété longhand.

Position de fond

La propriété `background-position` est utilisée pour spécifier la position de départ d'une image d'arrière-plan ou d'un dégradé

```
.myClass {
  background-image: url('path/to/image.jpg');
  background-position: 50% 50%;
}
```

La position est définie à l'aide des coordonnées **X** et **Y** et définie à l'aide de l'une des unités utilisées dans CSS.

Unité	La description
valeur % *valeur* %	Un pourcentage pour le décalage horizontal est relatif à *(largeur de la zone de positionnement d'arrière-plan - largeur de l'image d'arrière-plan)* . Un pourcentage pour le décalage vertical est relatif à *(hauteur de la zone de positionnement d'arrière-plan - hauteur de l'image d'arrière-plan)* La taille de l'image est la taille donnée par la taille de l' `background-size` .
valeur px *valeur* px	Décale l'image de fond d'une longueur donnée en pixels par rapport à la partie supérieure gauche de la zone de positionnement de l'arrière-plan

Les unités en CSS peuvent être spécifiées par différentes méthodes (voir ici).

Propriétés de la position d'arrière-plan à long terme

En plus de la propriété abrégée ci-dessus, on peut également utiliser les propriétés d' `background-position-x` long `background-position-x` et `background-position-y` . Celles-ci vous permettent de contrôler les positions x ou y séparément.

> **REMARQUE:** Ceci est pris en charge dans tous les navigateurs sauf Firefox (versions 31 à 48) 2 . Firefox 49, qui sera publié en septembre 2016, *prendra en* charge ces propriétés. Jusque-là, il existe un hack Firefox dans cette réponse Stack Overflow.

Attachement de fond

La propriété background-attachment définit si une image d'arrière-plan est fixe ou défile avec le reste de la page.

```
body {
  background-image: url('img.jpg');
  background-attachment: fixed;
}
```

Valeur	La description
faire défiler	L'arrière-plan défile avec l'élément. Ceci est la valeur par défaut.
fixé	L'arrière-plan est corrigé par rapport à la fenêtre d'affichage.
local	L'arrière-plan défile avec le contenu de l'élément.
initiale	Définit cette propriété sur sa valeur par défaut.
hériter	Hérite cette propriété de son élément parent.

Exemples

pièce jointe: défilement

Le comportement par défaut, lorsque le corps défile, l'arrière-plan défile avec:

```
body {
  background-image: url('image.jpg');
  background-attachment: scroll;
}
```

background-attachment: corrigé

L'image de fond sera fixe et ne bougera pas lorsque le corps défilera:

```
body {
  background-image: url('image.jpg');
  background-attachment: fixed;
}
```

background-attachment: local

L'image de fond du div défilera lorsque le contenu de la div défilera.

```
div {
  background-image: url('image.jpg');
  background-attachment: local;
}
```

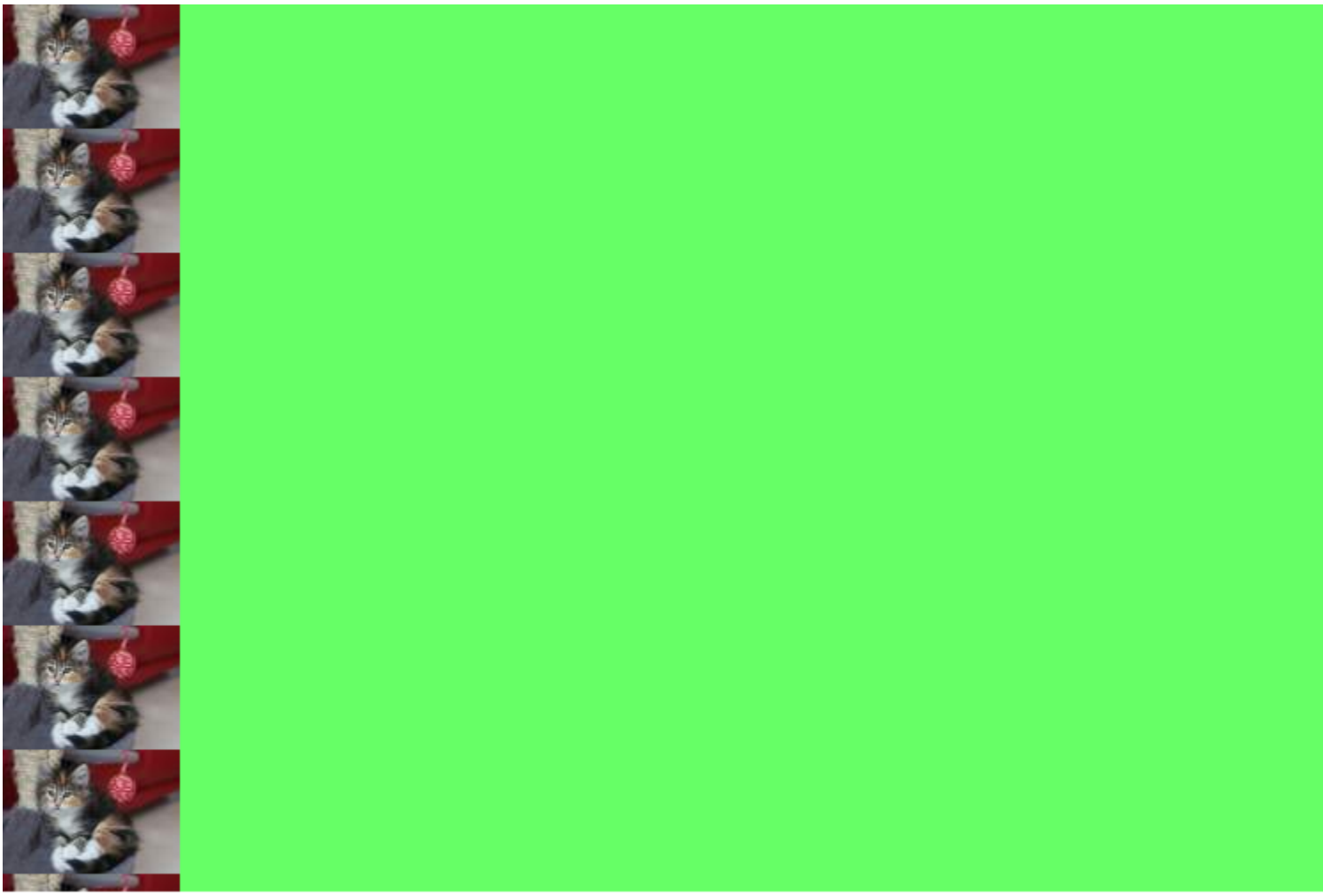
Répétition du fond

La propriété de répétition d'arrière-plan définit si / comment une image d'arrière-plan sera répétée.

Par défaut, une image d'arrière-plan est répétée verticalement et horizontalement.

```
div {
  background-image: url("img.jpg");
  background-repeat: repeat-y;
}
```

Voici à quoi ressemble une `background-repeat: repeat-y` :



Couleur de fond avec opacité

Si vous définissez l' `opacity` sur un élément, cela affectera tous ses éléments enfants. Pour définir une opacité juste sur l'arrière-plan d'un élément, vous devrez utiliser les couleurs RGBA. L'exemple suivant aura un arrière-plan noir avec une opacité de 0,6.

```
/* Fallback for web browsers that don't support RGBa */
background-color: rgb(0, 0, 0);

/* RGBa with 0.6 opacity */
background-color: rgba(0, 0, 0, 0.6);

/* For IE 5.5 - 7*/
filter: progid:DXImageTransform.Microsoft.gradient(startColorstr=#99000000,
endColorstr=#99000000);

/* For IE 8*/
-ms-filter: "progid:DXImageTransform.Microsoft.gradient(startColorstr=#99000000,
endColorstr=#99000000)";
```

Image d'arrière-plan multiple

En CSS3, nous pouvons empiler plusieurs arrière-plans dans le même élément.

```
#mydiv {
  background-image: url(img_1.png), /* top image */
                    url(img_2.png), /* middle image */
                    url(img_3.png); /* bottom image */
  background-position: right bottom,
                       left top,
                       right top;
  background-repeat: no-repeat,
                     repeat,
                     no-repeat;
}
```

Les images seront empilées les unes sur les autres avec le premier arrière-plan en haut et le dernier arrière-plan en arrière. `img_1` sera en haut, `img_2` et `img_3` en bas.

Nous pouvons également utiliser la propriété de raccourci d'arrière-plan pour ceci:

```
#mydiv {
  background: url(img_1.png) right bottom no-repeat,
              url(img_2.png) left top repeat,
              url(img_3.png) right top no-repeat;
}
```

Nous pouvons également empiler des images et des dégradés:

```
#mydiv {
  background: url(image.png) right bottom no-repeat,
              linear-gradient(to bottom, #fff 0%,#000 100%);
}
```

- Démo

La propriété background-origin

La propriété background-origin spécifie l'emplacement de l'image d'arrière-plan.

Remarque: Si la propriété `background-attachment` est définie sur `fixed` , cette propriété n'a aucun effet.

Valeur par défaut: `padding-box`

Valeurs possibles:

- `padding-box` - La position est relative à la boîte de rembourrage
- `border-box` - La position est relative à la case de bordure
- `content-box` - La position est relative à la zone de contenu
- `initial`
- `inherit`

CSS

```
.example {
  width: 300px;
  border: 20px solid black;
  padding: 50px;
  background: url(https://static.pexels.com/photos/6440/magazines-desk-work-workspace-
medium.jpg);
  background-repeat: no-repeat;
}

.example1 {}

.example2 { background-origin: border-box; }

.example3 { background-origin: content-box; }
```

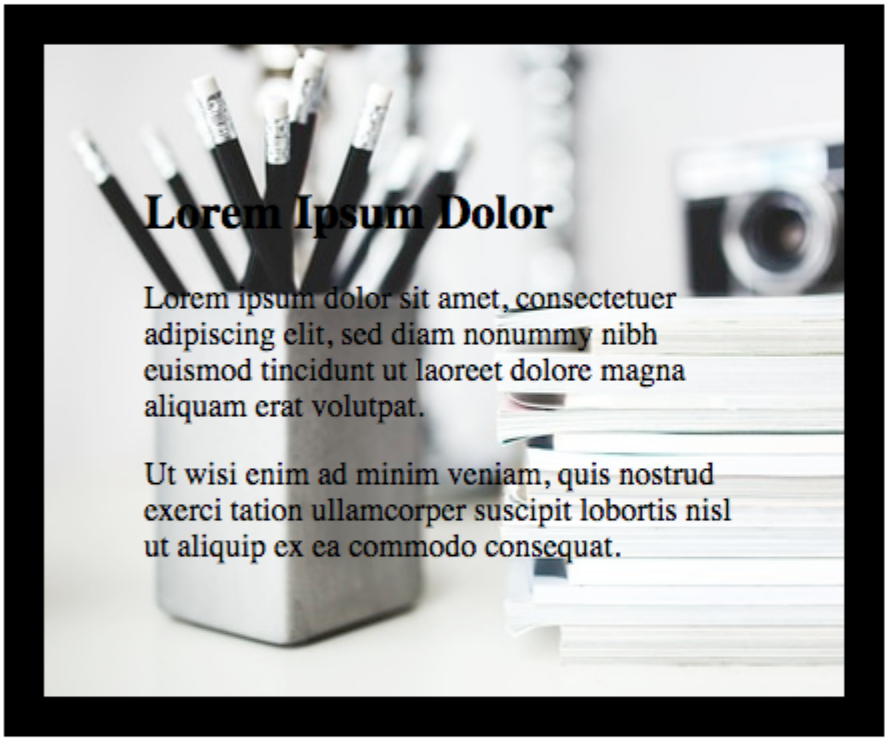
HTML

```
<p>No background-origin (padding-box is default):</p>

<div class="example example1">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: border-box:</p>
<div class="example example2">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: content-box:</p>
<div class="example example3">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>
```

Résultat:

No background-origin (padding-box is default):

background-origin: border-box:



background-origin: content-box:

est la valeur par défaut. Cela permet à l'arrière-plan de s'étendre jusqu'au bord extérieur de la bordure de l'élément.

- `padding-box` coupe l'arrière-plan au bord extérieur du rembourrage de l'élément et ne le laisse pas s'étendre jusqu'à la bordure;
- `content-box` coupe l'arrière-plan au bord de la zone de contenu.
- `inherit` applique le paramètre du parent à l'élément sélectionné.

CSS

```
.example {
  width: 300px;
  border: 20px solid black;
  padding: 50px;
  background: url(https://static.pexels.com/photos/6440/magazines-desk-work-workspace-
medium.jpg);
  background-repeat: no-repeat;
}

.example1 {}

.example2 { background-origin: border-box; }

.example3 { background-origin: content-box; }
```

HTML

```
<p>No background-origin (padding-box is default):</p>

<div class="example example1">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: border-box:</p>
<div class="example example2">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>

<p>background-origin: content-box:</p>
<div class="example example3">
  <h2>Lorem Ipsum Dolor</h2>
  <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh euismod
tincidunt ut laoreet dolore magna aliquam erat volutpat.</p>
  <p>Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis
nisl ut aliquip ex ea commodo consequat.</p>
</div>
```

Taille de fond

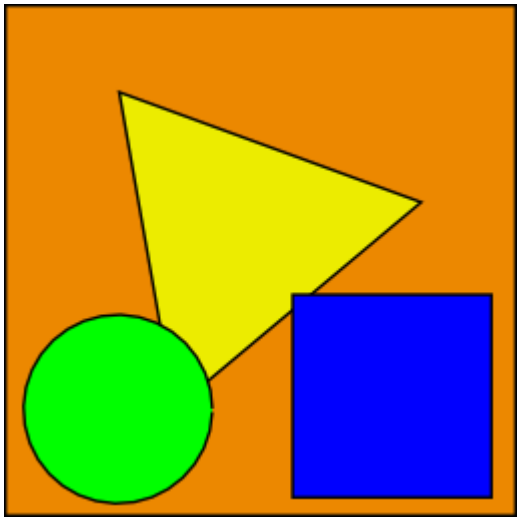
Aperçu général

La propriété `background-size` permet de contrôler la mise à l'échelle de l' `background-image` . Il prend jusqu'à deux valeurs, qui déterminent l'échelle / la taille de l'image résultante dans le sens vertical et horizontal. Si la propriété est manquante, son `auto` réputé en `width` et en `height` .

`auto` conservera le format de l'image, s'il peut être déterminé. La hauteur est facultative et peut être considérée comme `auto` . Par conséquent, sur une image de 256 px × 256 px, tous les paramètres de `background-size` suivants donneraient une image avec une hauteur et une largeur de 50 px:

```
background-size: 50px;
background-size: 50px auto; /* same as above */
background-size: auto 50px;
background-size: 50px 50px;
```

Donc, si nous avons commencé avec l'image suivante (qui a la taille mentionnée de 256 px × 256 px),

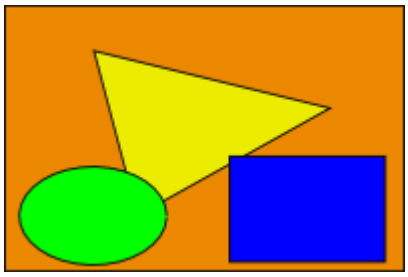


nous allons nous retrouver avec 50 px x 50 px sur l'écran de l'utilisateur, à l'arrière-plan de notre élément:

On peut également utiliser des valeurs de pourcentage pour redimensionner l'image en fonction de l'élément. L'exemple suivant donnerait une image dessinée de 200 px × 133 px:

```
#withbackground {
    background-image: url(to/some/background.png);

    background-size: 100% 66%;

    width: 200px;
    height: 200px;
```

```
    padding: 0;
    margin: 0;
}
```



Le comportement dépend de l' `background-origin` l' `background-origin` .

Garder le ratio d'aspect

Le dernier exemple de la section previos a perdu son format d'origine. Le cercle entra dans une ellipse, le carré dans un rectangle, le triangle dans un autre triangle.

L'approche de la longueur ou du pourcentage n'est pas assez flexible pour conserver le rapport d'aspect en tout temps. `auto` n'aide pas, car vous pourriez ne pas savoir quelle dimension de votre élément sera la plus grande. Cependant, pour couvrir complètement certaines zones avec une image (et corriger le format d'image) ou pour contenir une image avec un rapport d'aspect correct complètement dans une zone d'arrière-plan, les valeurs, le `contain` et la `cover` fournissent des fonctionnalités supplémentaires.

Eggsplanation pour `contain` et `cover`

Désolé pour le mauvais jeu de mots, mais nous allons utiliser une photo du jour de Biswarup Ganguly pour la démonstration. Disons qu'il s'agit de votre écran et que la zone grise est en dehors de votre écran visible. Pour démonstration, nous allons prendre un ratio de 16 × 9.

Nous voulons utiliser l'image ci-dessus de la journée comme arrière-plan. Cependant, nous avons

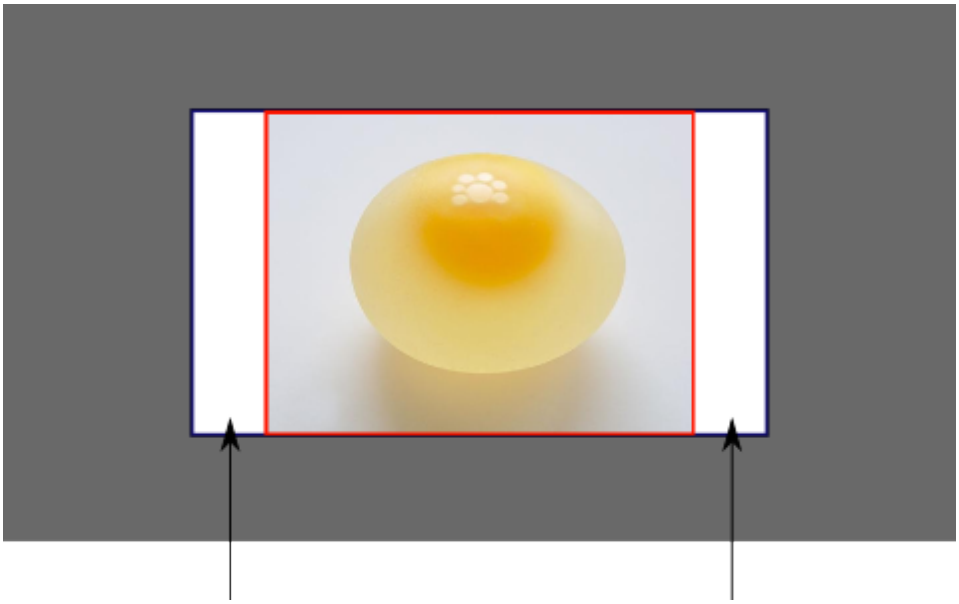
recadré l'image en 4x3 pour une raison quelconque. Nous pourrions définir une propriété de `background-size` à une longueur fixe, mais nous nous concentrerons sur le `contain` et la `cover` . Notez que je suppose également que nous n'avons pas modifié la largeur et / ou la hauteur du `body` .

contain

```
contain
```

> Mettez à l'échelle l'image, tout en conservant son rapport d'aspect intrinsèque (le cas échéant), à la plus grande taille, de sorte que sa largeur et sa hauteur puissent tenir dans la zone de positionnement de l'arrière-plan.

Cela garantit que l'image d'arrière-plan est toujours complètement contenue dans la zone de positionnement de l'arrière-plan. Cependant, il pourrait y avoir un espace vide rempli de votre `background-color` dans ce cas:

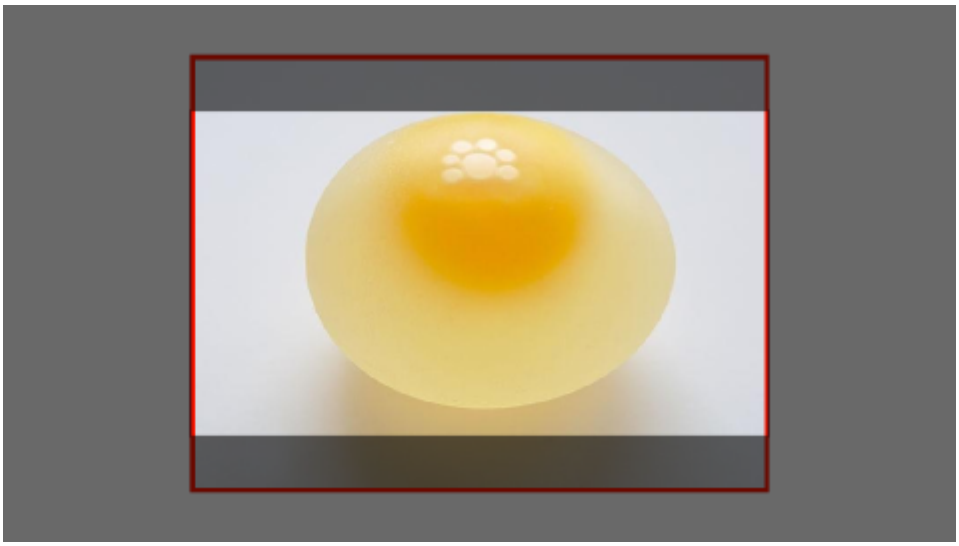


cover

```
cover
```

> Mettez à l'échelle l'image, tout en conservant son rapport d'aspect intrinsèque (le cas échéant), à la plus petite taille, de sorte que sa largeur et sa hauteur puissent couvrir complètement la zone de positionnement de l'arrière-plan.

Cela garantit que l'image de fond couvre tout. Il n'y aura pas de `background-color` visible, mais en fonction du ratio de l'écran, une grande partie de votre image pourrait être coupée:



Démonstration avec le code actuel

```
div > div {
  background-image: url(http://i.stack.imgur.com/r5CAq.jpg);
  background-repeat: no-repeat;
  background-position: center center;
  background-color: #ccc;
  border: 1px solid;
  width: 20em;
  height: 10em;
}
div.contain {
  background-size: contain;
}
div.cover {
  background-size: cover;
}
/*****************************************
 Additional styles for the explanation boxes
*****************************************/

div > div {
  margin: 0 1ex 1ex 0;
  float: left;
}
div + div {
  clear: both;
  border-top: 1px dashed silver;
  padding-top:1ex;
}
div > div::after {
  background-color: #000;
  color: #fefefe;
  margin: 1ex;
  padding: 1ex;
  opacity: 0.8;
  display: block;
  width: 10ex;
  font-size: 0.7em;
  content: attr(class);
}
```

```
<div>
  <div class="contain"></div>
  <p>Note the grey background. The image does not cover the whole region, but it's fully
<em>contained</em>.
  </p>
</div>
<div>
  <div class="cover"></div>
  <p>Note the ducks/geese at the bottom of the image. Most of the water is cut, as well as a
part of the sky. You don't see the complete image anymore, but neither do you see any
background color; the image <em>covers</em> all of the <code>&lt;div&gt;</code>.</p>
</div>
```

Note the grey background. The image does not cover the whole region, but it's fully *contained*.

Note the ducks/geese at the bottom of the image. Most of the water is cut, as well as a part of the sky. You don't see the complete image anymore, but neither do you see any background color; the image *covers* all of the `<div>`.

Propriété background-blend-mode

```
.my-div {
    width: 300px;
    height: 200px;
    background-size: 100%;
    background-repeat: no-repeat;
    background-image: linear-gradient(to right, black 0%,white 100%),
url('https://static.pexels.com/photos/54624/strawberry-fruit-red-sweet-54624-medium.jpeg');
    background-blend-mode:saturation;
}
```

```
<div class="my-div">Lorem ipsum</div>
```

Voir le résultat ici: https://jsfiddle.net/MadalinaTn/y69d28Lb/

Syntaxe CSS: background-blend-mode: normal | multiplier | écran | superposition | assombrir | éclaircir | couleur-dodge | saturation | couleur | luminosité;

Lire Arrière-plans en ligne: https://riptutorial.com/fr/css/topic/296/arriere-plans

Chapitre 4: boîte ombre

Syntaxe

- boîte-ombre: aucune | h-ombre v-ombre flou étalement couleur | inset | initial | hérite;

Paramètres

Paramètres	Détails
encart	par défaut, l'ombre est traitée comme une ombre portée. le mot-clé encart dessine l'ombre à l'intérieur du cadre / de la bordure.
offset-x	la distance horizontale
offset-y	la distance verticale
rayon de flou	0 par défaut. la valeur ne peut pas être négative. plus la valeur est grande, plus l'ombre devient grande et légère.
rayon de propagation	0 par défaut. les valeurs positives entraîneront l'expansion de l'ombre. les valeurs négatives réduiront l'ombre.
Couleur	peut être de différentes notations: un mot-clé couleur, hexadécimal, `rgb()` , `rgba()` , `hsl()` , `hsla()`

Remarques

Support du navigateur:

- Chrome 10.0
- IE 9.0
- Firefox 4.0 3.5-moz
- Safari 5.1 3.1 -webkit-
- Opera 10.5

Examples

ombre portée

JSFiddle: https://jsfiddle.net/UnsungHero97/80qod7aL/

HTML

```
<div class="box_shadow"></div>
```

CSS

```
.box_shadow {
  -webkit-box-shadow: 0px 0px 10px -1px #444444;
     -moz-box-shadow: 0px 0px 10px -1px #444444;
          box-shadow: 0px 0px 10px -1px #444444;
}
```

ombre portée intérieure

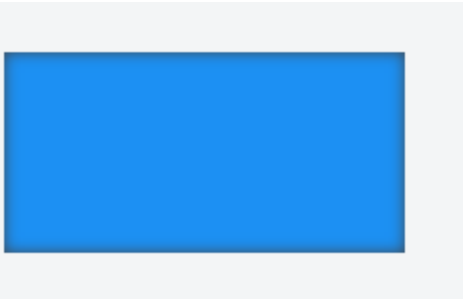
HTML

```
<div class="box_shadow"></div>
```

CSS

```
.box_shadow {
  background-color: #1C90F3;
  width: 200px;
  height: 100px;
  margin: 50px;
  -webkit-box-shadow: inset 0px 0px 10px 0px #444444;
     -moz-box-shadow: inset 0px 0px 10px 0px #444444;
          box-shadow: inset 0px 0px 10px 0px #444444;
}
```

Résultat:



JSFiddle: https://jsfiddle.net/UnsungHero97/80qod7aL/1/

Ombre portée inférieure uniquement à l'aide d'un pseudo-élément

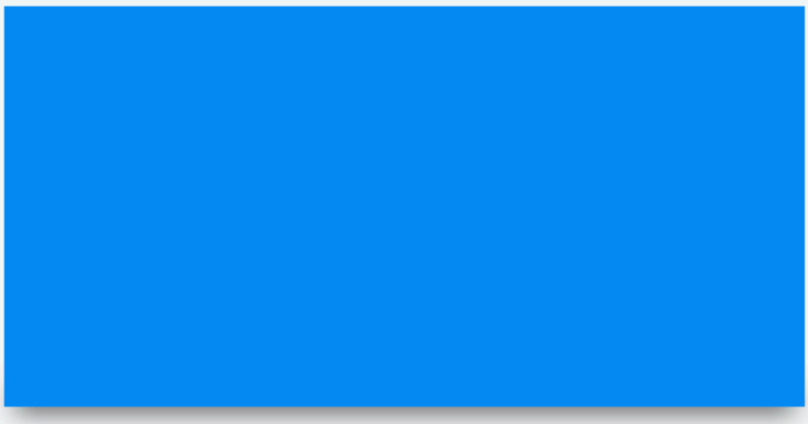
JSFiddle: https://jsfiddle.net/UnsungHero97/80qod7aL/2/

HTML

```
<div class="box_shadow"></div>
```

CSS

```
.box_shadow {
  background-color: #1C90F3;
  width: 200px;
  height: 100px;
  margin: 50px;
}

.box_shadow:after {
  content: "";
  width: 190px;
  height: 1px;
  margin-top: 98px;
  margin-left: 5px;
  display: block;
  position: absolute;
  z-index: -1;
  -webkit-box-shadow: 0px 0px 8px 2px #444444;
     -moz-box-shadow: 0px 0px 8px 2px #444444;
          box-shadow: 0px 0px 8px 2px #444444;
}
```



plusieurs ombres

JSFiddle: https://jsfiddle.net/UnsungHero97/80qod7aL/5/

HTML

```
<div class="box_shadow"></div>
```

CSS

```
.box_shadow {
  width: 100px;
  height: 100px;
```

```
  margin: 100px;
  box-shadow:
    -52px -52px 0px 0px #f65314,
    52px -52px 0px 0px #7cbb00,
    -52px 52px 0px 0px #00a1f1,
    52px 52px 0px 0px #ffbb00;
}
```

Lire boîte ombre en ligne: https://riptutorial.com/fr/css/topic/1746/boite-ombre

Chapitre 5: Cascade et spécificité

Remarques

La spécificité CSS vise à favoriser la concision du code en permettant à un auteur de définir des règles de mise en forme générales pour un large ensemble d'éléments, puis de les remplacer pour un certain sous-ensemble.

Examples

En cascade

La cascade et la spécificité sont utilisées ensemble pour déterminer la valeur finale d'une propriété de style CSS. Ils définissent également les mécanismes de résolution des conflits dans les jeux de règles CSS.

Ordre de chargement CSS

Les styles sont lus à partir des sources suivantes, dans cet ordre:

1. Feuille de style de l'agent utilisateur (styles fournis par le fournisseur du navigateur)
2. Feuille de style utilisateur (le style supplémentaire défini par l'utilisateur sur son navigateur)
3. Feuille de style de l'auteur (Auteur signifie le créateur de la page Web / du site Web)
    - Peut-être un ou plusieurs fichiers `.css`
    - Dans l'élément `<style>` du document HTML
4. Styles en ligne (dans l'attribut de `style` sur un élément HTML)

Le navigateur recherchera les styles correspondants lors du rendu d'un élément.

Comment les conflits sont-ils résolus?

Lorsqu'un seul ensemble de règles CSS tente de définir un style pour un élément, il n'y a pas de conflit et ce jeu de règles est utilisé.

Lorsque plusieurs jeux de règles sont détectés avec des paramètres en conflit, commencez par définir les règles de spécification, puis les règles en cascade pour déterminer le style à utiliser.

Exemple 1 - Règles de spécificité

```
.mystyle { color: blue; }  /* specificity: 0, 0, 1, 0 */
div { color: red; }        /* specificity: 0, 0, 0, 1 */
```

```
<div class="mystyle">Hello World</div>
```

Quelle couleur sera le texte? (survolez pour voir la réponse)

> bleu

Tout d'abord, les règles de spécificité sont appliquées et celle qui présente la plus grande spécificité "gagne".

Exemple 2 - Règles en cascade avec sélecteurs identiques

Fichier css externe

```
.class {
  background: #FFF;
}
```

CSS interne (en fichier HTML)

```
<style>
.class {
  background: #000;
}
<style>
```

Dans ce cas, lorsque vous avez des sélecteurs identiques, la cascade intervient et détermine que le dernier chargé "gagne".

Exemple 3 - Règles en cascade après les règles de spécificité

```
body > .mystyle { background-color: blue; }   /* specificity: 0, 0, 1, 1 */
.otherstyle > div { background-color: red; }  /* specificity: 0, 0, 1, 1 */
```

```
<body class="otherstyle">
  <div class="mystyle">Hello World</div>
</body>
```

De quelle couleur sera l'arrière-plan?

> rouge

Après avoir appliqué les règles de spécificité, il existe toujours un conflit entre le bleu et le rouge, de sorte que les règles en cascade sont appliquées en plus des règles de spécificité. La mise en cascade examine l'ordre de chargement des règles, que ce soit dans le même fichier `.css` ou dans la collection de sources de style. Le dernier chargé remplace les précédents. Dans ce cas, la règle `.otherstyle > div` "gagne".

Une note finale

- La spécificité du sélecteur prime toujours.
- Les feuilles de style ordonnent les cravates.
- Les styles en ligne l'emportent sur tout.

La déclaration! Importante

La déclaration `!important` est utilisée pour remplacer la spécificité habituelle dans une feuille de style en donnant une priorité plus élevée à une règle. Son usage est: `property : value !important;`

```
#mydiv {
    font-weight: bold !important;    /* This property won't be overridden
                                        by the rule below */
}

#outerdiv #mydiv {
    font-weight: normal;            /* #mydiv font-weight won't be set to normal
                                    even if it has a higher specificity because
                                    of the !important declaration above */
}
```

Eviter l'utilisation de `!important` est fortement recommandé (sauf en cas d'absolue nécessité), car cela perturbera le flux naturel des règles CSS qui peuvent entraîner une incertitude dans votre feuille de style. En outre, il est important de noter que lorsque plusieurs déclarations `!important` sont appliquées à la même règle sur un élément donné, la plus spécifique sera celle qui sera appliquée.

Voici quelques exemples où l'utilisation de `!important` Déclaration `!important` peut être justifiée:

- Si vos règles ne doivent pas être remplacées par un style en ligne de l'élément écrit dans l'attribut de `style` de l'élément html.
- Pour donner à l'utilisateur plus de contrôle sur l'accessibilité du Web, comme augmenter ou diminuer la taille de la taille de la police, en remplaçant le style d'auteur par `!important` .
- Pour tester et déboguer en utilisant l'élément inspect.

Voir également:

- **W3C - 6 Affectation de valeurs de propriété, de cascade et d'héritage - 6.4.2! Règles importantes**

Calcul de la spécificité du sélecteur

Chaque sélecteur CSS individuel a sa propre valeur de spécificité. Chaque sélecteur dans une séquence augmente la spécificité globale de la séquence. Les sélecteurs appartiennent à l'un des trois groupes de spécificités suivants: *A* , *B* et *c* . Lorsque plusieurs séquences de sélecteur sélectionnent un élément donné, le navigateur utilise les styles appliqués par la séquence avec la spécificité globale la plus élevée.

Groupe	Composé de	Exemples
UNE	sélecteurs d'identification	`#foo`

B	sélecteurs de classe
	sélecteurs d'attribut
	pseudo-classes
		`.bar`
		`[title]` , `[colspan="2"]`
		`:hover` `:nth-child(2)`

c	sélecteurs de type pseudo-éléments	`div` , `li` `::before` , `::first-letter`

Le groupe *A* est le plus spécifique, suivi du groupe *B* , puis enfin du groupe *c* .

Le sélecteur universel (`*`) et les combinateurs (comme `>` et `~`) n'ont pas de spécificité.

Exemple 1: Spécificité de différentes séquences de sélecteurs

```
#foo #baz {}      /* a=2, b=0, c=0 */

#foo.bar {}       /* a=1, b=1, c=0 */

#foo {}           /* a=1, b=0, c=0 */

.bar:hover {}     /* a=0, b=2, c=0 */

div.bar {}        /* a=0, b=1, c=1 */

:hover {}         /* a=0, b=1, c=0 */

[title] {}        /* a=0, b=1, c=0 */

.bar {}           /* a=0, b=1, c=0 */

div ul + li {}    /* a=0, b=0, c=3 */

p::after {}       /* a=0, b=0, c=2 */

*::before {}      /* a=0, b=0, c=1 */

::before {}       /* a=0, b=0, c=1 */

div {}            /* a=0, b=0, c=1 */

* {}              /* a=0, b=0, c=0 */
```

Exemple 2: Comment la spécificité est utilisée par le navigateur

Imaginez l'implémentation CSS suivante:

```
#foo {
  color: blue;
}

.bar {
  color: red;
```

```
  background: black;
}
```

Nous avons ici un sélecteur d'ID qui déclare la `color` en *bleu* et un sélecteur de classe qui déclare la `color` en *rouge* et l' `background` - `background` en *noir* .

Un élément avec un ID de `#foo` et une classe de `.bar` sera sélectionné par les deux déclarations. Les sélecteurs d'ID ont une spécificité pour le groupe *A* et les sélecteurs de classe ont une spécificité pour le groupe *B.* Un sélecteur d'ID l'emporte sur un nombre quelconque de sélecteurs de classe. Pour cette raison, `color:blue;` depuis le sélecteur `#foo` et l' `background:black;` - `background:black;` du sélecteur `.bar` sera appliqué à l'élément. La plus grande spécificité du sélecteur d'ID entraînera le navigateur à ignorer la `.bar` de `color` du sélecteur `.bar` .

Imaginez maintenant une implémentation CSS différente:

```
.bar {
  color: red;
  background: black;
}

.baz {
  background: white;
}
```

Ici, nous avons deux sélecteurs de classe; l'un d'eux déclare la `color` en *rouge* et l' `background` - `background` en *noir* et l'autre déclare l' `background` en *blanc* .

Un élément à la fois avec les classes `.bar` et `.baz` sera affecté par ces deux déclarations, mais le problème actuel est que les deux `.bar` et `.baz` ont une spécificité de groupe *B* identique. La nature en cascade de CSS résout ce problème pour nous: comme `.baz` est défini *après* `.bar` , notre élément se retrouve avec la `color` *rouge* de `.bar` mais le `background` *blanc* de `.baz` .

Exemple 3: Comment manipuler la spécificité

Le dernier extrait de l'exemple 2 ci-dessus peut être manipulé pour garantir que la déclaration de `color` notre sélecteur de classe `.bar` est utilisée à la place de celle du sélecteur de classe `.baz` .

```
.bar {}         /* a=0, b=1, c=0 */
.baz {}         /* a=0, b=1, c=0 */
```

La manière la plus courante d'y parvenir serait de savoir quels autres sélecteurs peuvent être appliqués à la séquence de sélecteur `.bar` . Par exemple, si la `.bar` classe ne fut jamais appliquée pour `span` des éléments, nous pourrions modifier le `.bar` sélecteur `span.bar` . Cela lui donnerait une nouvelle spécificité du groupe *C* , qui outrepasserait le `.baz` sélecteur `.baz` :

```
span.bar {}     /* a=0, b=1, c=1 */
.baz {}         /* a=0, b=1, c=0 */
```

Cependant, il n'est pas toujours possible de trouver un autre sélecteur commun qui est partagé

entre tous les éléments utilisant la classe `.bar` . Pour cette raison, CSS nous permet de dupliquer les sélecteurs pour augmenter la spécificité. Au lieu de simplement `.bar` , nous pouvons utiliser `.bar.bar` place (Voir La grammaire des sélecteurs, Recommandation du W3C). Cela sélectionne toujours n'importe quel élément avec une classe de `.bar` , mais a maintenant le double de la spécificité du groupe *B* :

```
.bar.bar {}     /* a=0, b=2, c=0 */
.baz {}         /* a=0, b=1, c=0 */
```

`!important` déclarations de style `!important` et en ligne

Le drapeau `!important` sur une déclaration de style et les styles déclarés par l'attribut de `style` HTML sont considérés comme ayant une plus grande spécificité que n'importe quel sélecteur. Si elles existent, la déclaration de style qu'elles affectent remplacera les autres déclarations, quelle que soit leur spécificité. C'est-à-dire, sauf si vous avez plus d'une déclaration contenant un indicateur `!important` pour la même propriété qui s'applique au même élément. Ensuite, les règles de spécificité normales s'appliqueront à ces propriétés en référence les unes aux autres.

Parce qu'ils remplacent complètement la spécificité, l'utilisation de `!important` est mal vue dans la plupart des cas d'utilisation. On devrait l'utiliser le moins possible. Pour que le code CSS soit efficace et maintenable à long terme, il est presque toujours préférable d'augmenter la spécificité du sélecteur environnant plutôt que d'utiliser `!important` .

Une de ces rares exceptions où `!important` n'est pas mal vu, est l'implémentation de classes d'assistance génériques comme une `.hidden` ou `.background-yellow` censées toujours remplacer une ou plusieurs propriétés où qu'elles soient rencontrées. Et même alors, vous devez savoir ce que vous faites. La dernière chose que vous voulez, lorsque vous écrivez des CSS maintenables, est d'avoir `!important` indicateurs `!important` dans votre CSS.

Une note finale

Une idée fausse commune à propos de la spécificité des CSS est que les valeurs des groupes *A* , *B* et *c* doivent être combinées entre elles (`a=1, b=5, c=1` => 151). Ce n'est **pas** le cas. Si tel était le cas, il serait suffisant d'avoir 20 de sélecteur de groupe *B* ou *c* pour remplacer un seul sélecteur de groupe *A* ou *B* , respectivement. Les trois groupes doivent être considérés comme des niveaux de spécificité individuels. La spécificité ne peut être représentée par une seule valeur.

Lors de la création de votre feuille de style CSS, vous devez conserver la spécificité la plus faible possible. Si vous devez améliorer la spécificité pour remplacer une autre méthode, augmentez-la, mais aussi faible que possible pour la rendre plus élevée. Vous ne devriez pas avoir besoin d'un sélecteur comme celui-ci:

```
body.page header.container nav div#main-nav li a {}
```

Cela rend les changements futurs plus difficiles et pollue cette page CSS.

Vous pouvez calculer la spécificité de votre sélecteur ici

Exemple de spécificité plus complexe

```
div {
    font-size: 7px;
    border: 3px dotted pink;
    background-color: yellow;
    color: purple;
}

body.mystyle > div.myotherstyle {
    font-size: 11px;
    background-color: green;
}

#elmnt1 {
    font-size: 24px;
    border-color: red;
}

.mystyle .myotherstyle {
    font-size: 16px;
    background-color: black;
    color: red;
}
```

```
<body class="mystyle">
    <div id="elmnt1" class="myotherstyle">
        Hello, world!
    </div>
</body>
```

Quelles frontières, couleurs et tailles de police le texte sera-t-il?

taille de police:

> `font-size: 24; #elmnt1` jeu de règles `#elmnt1` a la spécificité la plus élevée pour le `<div>` en question, chaque propriété est définie ici.

frontière:

> `border: 3px dotted red;` . La couleur de bordure `red` est extraite du `#elmnt1` règles `#elmnt1` , car elle possède la plus grande spécificité. Les autres propriétés de la bordure, de l'épaisseur de la bordure et du style de bordure proviennent du jeu de règles `div` .

Couleur de fond:

> `background-color: green;` . La `background-color` est définie dans les `body.mystyle > div.myotherstyle div` , `body.mystyle > div.myotherstyle` et `.mystyle .myotherstyle` . Les spécificités sont (0, 0, 1) vs (0, 2, 2) vs (0, 2, 0), donc le milieu "gagne".

Couleur:

> `color: red;` . La couleur est définie à la fois dans les jeux de règles `div` et `.mystyle`

`.myotherstyle` . Ce dernier a la spécificité la plus élevée de (0, 2, 0) et "gagne".

Lire Cascade et spécificité en ligne: https://riptutorial.com/fr/css/topic/450/cascade-et-specificite

Chapitre 6: Centrage

Examples

Utiliser la transformation CSS

Les transformations CSS sont basées sur la taille des éléments, donc si vous ne savez pas quelle est la taille ou la largeur de votre élément, vous pouvez le positionner à 50% du haut et de la gauche d'un conteneur relatif et le traduire de 50% à gauche le centrer verticalement et horizontalement.

Gardez à l'esprit qu'avec cette technique, l'élément pourrait être rendu à une limite de pixels non entiers, ce qui le rendrait flou. Voir cette réponse dans SO pour une solution de contournement.

HTML

```
<div class="container">
  <div class="element"></div>
</div>
```

CSS

```
.container {
  position: relative;
}

.element {
  position: absolute;
  top: 50%;
  left: 50%;
  transform: translate(-50%, -50%);
}
```

Voir l'exemple dans JSFiddle

COMPATIBILITÉ DU NAVIGATEUR CROISÉ

La propriété transform nécessite des préfixes à prendre en charge par les anciens navigateurs. Des préfixes sont nécessaires pour Chrome <= 35, Safari <= 8, Opera <= 22, Android Browser <= 4.4.4 et IE9. Les transformations CSS ne sont pas prises en charge par IE8 et les versions antérieures.

Voici une déclaration de transformation commune à l'exemple précédent:

```
-webkit-transform: translate(-50%, -50%); /* Chrome, Safari, Opera, Android */
    -ms-transform: translate(-50%, -50%); /* IE 9 */
        transform: translate(-50%, -50%);
```

Pour plus d'informations, voir canIuse .

PLUS D'INFORMATION

- L'élément est positionné en fonction du premier parent non statique (`position: relative` , `absolute` ou `fixed`). Explorez plus dans ce violon et ce sujet de documentation .
- Pour le centrage horizontal uniquement, utilisez à `left: 50%` et `transform: translateX(-50%)` . Il en va de même pour le centrage vertical uniquement: centre avec `top: 50%` et `transform: translateY(-50%)` .
- L'utilisation d'un élément largeur / hauteur non statique avec cette méthode de centrage peut entraîner un écrasement de l'élément centré. Cela se produit principalement avec des éléments contenant du texte, et peut être corrigé en ajoutant: `margin-right: -50%;` et `margin-bottom: -50%;` . Voir ce violon pour plus d'informations.

Utiliser Flexbox

HTML:

```
<div class="container">
  <img src="http://lorempixel.com/400/200" />
</div>
```

CSS:

```
html, body, .container {
  height: 100%;
}
.container {
  display: flex;
  justify-content: center; /* horizontal center */
}
img {
  align-self: center; /* vertical center */
}
```

Voir résultat

HTML:

```
<img src="http://lorempixel.com/400/200" />
```

CSS:

```
html, body {
  height: 100%;
}
body {
  display: flex;
```

```
  justify-content: center; /* horizontal center */
  align-items: center;     /* vertical center */
}
```

Voir résultat

Reportez-vous à la section Centrage vertical et horizontal dynamique sous la documentation Flexbox pour plus de détails sur flexbox et leur signification.

Prise en charge du navigateur

Flexbox est pris en charge par tous les principaux navigateurs, à l' exception des versions d'IE antérieures à 10 .

Certaines versions de navigateur récentes, telles que Safari 8 et IE10, nécessitent des préfixes de fournisseur .

Pour un moyen rapide de générer des préfixes, il existe Autoprefixer , un outil tiers.

Pour les navigateurs plus anciens (comme IE 8 et 9), un Polyfill est disponible .

Pour un aperçu plus détaillé de la prise en charge du navigateur Flexbox, consultez cette réponse .

Utilisation de la position: absolue

Travailler dans les anciens navigateurs (IE> = 8)

Les marges automatiques, associées à des valeurs nulles pour les décalages `left` et `right` ou `top` et `bottom` centreront les éléments positionnés de manière absolue dans leur parent.

Voir résultat

HTML

```
<div class="parent">
  <img class="center" src="http://lorempixel.com/400/200/" />
</div>
```

CSS

```
.parent {
  position: relative;
  height: 500px;
}

.center {
  position: absolute;
  margin: auto;
  top: 0;
  right: 0;
  bottom: 0;
```

```
  left: 0;
}
```

Les éléments qui n'ont pas leur propre largeur et hauteur implicites, comme le font les images, auront besoin de ces valeurs définies.

Autres ressources: Centrage absolu en CSS

Technique d'élément fantôme (hack de Michał Czernow)

Cette technique fonctionne même lorsque les dimensions du conteneur sont inconnues.

Configurez un élément "fantôme" à l'intérieur du conteneur à centrer en hauteur, puis utilisez `vertical-align: middle` à la fois sur celui-ci et sur l'élément à centrer.

CSS

```
/* This parent can be any width and height */
.block {
  text-align: center;

  /* May want to do this if there is risk the container may be narrower than the element
inside */
  white-space: nowrap;
}

/* The ghost element */
.block:before {
  content: '';
  display: inline-block;
  height: 100%;
  vertical-align: middle;

  /* There is a gap between ghost element and .centered,
  caused by space character rendered. Could be eliminated by
  nudging .centered (nudge distance depends on font family),
  or by zeroing font-size in .parent and resetting it back
  (probably to 1rem) in .centered. */
  margin-right: -0.25em;
}

/* The element to be centered, can also be of any width and height */
.centered {
  display: inline-block;
  vertical-align: middle;
  width: 300px;
  white-space: normal; /* Resetting inherited nowrap behavior */
}
```

HTML

```
<div class="block">
  <div class="centered"></div>
</div>
```

Utiliser text-align

Le type de centrage le plus courant et le plus simple est celui des lignes de texte dans un élément. CSS a la règle `text-align: center` à cette fin:

HTML

```
<p>Lorem ipsum</p>
```

CSS

```
p {
    text-align: center;
}
```

Cela ne fonctionne pas pour centrer des éléments de bloc entiers . `text-align` contrôle uniquement l'alignement du contenu en ligne comme le texte dans son élément de bloc parent.

Voir plus sur l' `text-align` dans la section Typographie .

Centrage par rapport à un autre élément

Nous verrons comment centrer le contenu en fonction de la hauteur d'un élément proche.

Compatibilité: IE8 +, tous les autres navigateurs modernes.

HTML

```
<div class="content">
  <div class="position-container">
    <div class="thumb">
      <img src="http://lorempixel.com/400/200/">
    </div>
    <div class="details">
      <p class="banner-title">text 1</p>
      <p class="banner-text">content content content content content content content content
content content content content content content</p>
      <button class="btn">button</button>
    </div>
  </div>
</div>
```

CSS

```
.content * {
  box-sizing: border-box;
}
.content .position-container {
  display: table;
}
.content .details {
  display: table-cell;
  vertical-align: middle;
```

```
  width: 33.333333%;
  padding: 30px;
  font-size: 17px;
  text-align: center;
}
.content .thumb {
  width: 100%;
}
.content .thumb img {
  width: 100%;
}
```

Lien vers JSFiddle

Les principaux points sont les 3 `.thumb` , `.details` et `.position-container` container:

- Le `.position-container` doit avoir `display: table` .
- Les `.details` doivent avoir la largeur réelle définie en `width: ....` et `display: table-cell` , `vertical-align: middle` .
- Le `.thumb` doit avoir la `width: 100%` si vous voulez qu'il prenne tout l'espace restant et il sera influencé par la largeur `.details` .
- L'image (si vous avez une image) à l'intérieur de `.thumb` devrait avoir la `width: 100%` , mais ce n'est pas nécessaire si vous avez des proportions correctes.

Vertical aligner n'importe quoi avec 3 lignes de code

Soutenu par IE11 +

Voir résultat

Utilisez ces 3 lignes pour aligner verticalement pratiquement tout. Assurez-vous que la div / image que vous appliquez le code a un parent avec une hauteur.

CSS

```
div.vertical {
  position: relative;
  top: 50%;
  transform: translateY(-50%);
}
```

HTML

```
<div class="vertical">Vertical aligned text!</div>
```

Aligner verticalement une image à l'intérieur d'une div

HTML

```
<div class="wrap">
    <img src="http://lorempixel.com/400/200/" />
</div>
```

CSS

```
.wrap {
    height: 50px;/* max image height */
    width: 100px;
    border: 1px solid blue;
    text-align: center;
}
.wrap:before {
  content:"";
  display: inline-block;
  height: 100%;
  vertical-align: middle;
  width: 1px;
}

img {
    vertical-align: middle;
}
```

Centrage horizontal et vertical à l'aide d'une disposition de tableau

On pourrait facilement centrer un élément enfant en utilisant la propriété d'affichage de la `table` .

HTML

```
<div class="wrapper">
    <div class="parent">
        <div class="child"></div>
    </div>
</div>
```

CSS

```
.wrapper {
    display: table;
    vertical-align: center;
    width: 200px;
    height: 200px;
    background-color: #9e9e9e;
}
.parent {
    display: table-cell;
    vertical-align: middle;
    text-align: center;
}
.child {
    display: inline-block;
    vertical-align: middle;
    text-align: center;
    width: 100px;
    height: 100px;
```

```
    background-color: teal;
}
```

Utiliser calc ()

La fonction calc () est la partie d'une nouvelle syntaxe dans CSS3 dans laquelle vous pouvez calculer (mathématiquement) la taille / position de votre élément en utilisant diverses valeurs telles que les pixels, les pourcentages, etc. Remarque: - Lorsque vous utilisez cette fonction , prenez toujours soin de l'espace entre deux valeurs `calc(100% - 80px)` .

CSS

```
.center {
    position: absolute;
    height: 50px;
    width: 50px;
    background: red;
    top: calc(50% - 50px / 2); /* height divided by 2*/
    left: calc(50% - 50px / 2); /* width divided by 2*/
}
```

HTML

```
<div class="center"></div>
```

Aligner verticalement les éléments de hauteur dynamique

Appliquer intuitivement css ne produit pas les résultats souhaités car

- `vertical-align:middle` ***ne*** s'applique ***pas*** aux éléments de niveau bloc
- `margin-top:auto` et `margin-bottom:auto` valeurs utilisées `margin-bottom:auto` calculées sur ***zéro***
- `margin-top:-50%` valeurs de marge basées sur un pourcentage de `margin-top:-50%` sont calculées par rapport à la ***largeur*** du bloc contenant

Pour prendre en charge le navigateur le plus large, une solution de contournement avec les éléments d'aide:

HTML

```
<div class="vcenter--container">
  <div class="vcenter--helper">
    <div class="vcenter--content">
      <!--stuff-->
    </div>
  </div>
</div>
```

CSS

```
.vcenter--container {
  display: table;
```

```
  height: 100%;
  position: absolute;
  overflow: hidden;
  width: 100%;
}
.vcenter--helper {
  display: table-cell;
  vertical-align: middle;
}
.vcenter--content {
  margin: 0 auto;
  width: 200px;
}
```

jsfiddle de la question originale . Cette approche

- fonctionne avec des éléments de hauteur dynamiques
- respecte le flux de contenu
- est pris en charge par les navigateurs existants

Utilisation de la hauteur de ligne

Vous pouvez également utiliser la `line-height` de `line-height` pour centrer verticalement une seule ligne de texte dans un conteneur:

CSS

```
div {
    height: 200px;
    line-height: 200px;
}
```

C'est assez moche, mais peut être utile dans un élément `<input />` . La propriété `line-height` ne fonctionne que lorsque le texte à centrer couvre une seule ligne. Si le texte se divise en plusieurs lignes, le résultat obtenu ne sera pas centré.

Centrage vertical et horizontal sans se soucier de la hauteur ou de la largeur

La technique suivante vous permet d'ajouter votre contenu à un élément HTML et de le centrer horizontalement et verticalement **sans vous soucier de sa hauteur ou de sa largeur** .

Le récipient extérieur

- devrait avoir l' `display: table;`

Le récipient intérieur

- devrait avoir l' `display: table-cell;`
- devrait avoir `vertical-align: middle;`
- devrait avoir `text-align: center;`

La boîte de contenu

- devrait avoir l' `display: inline-block;`
- devrait réajuster l'alignement de texte horizontal sur, par exemple. `text-align: left;` ou `text-align: right;` , sauf si vous voulez que le texte soit centré

Démo

HTML

```
<div class="outer-container">
   <div class="inner-container">
     <div class="centered-content">
        You can put anything here!
     </div>
   </div>
</div>
```

CSS

```
body {
    margin : 0;
}

.outer-container {
    position : absolute;
    display: table;
    width: 100%; /* This could be ANY width */
    height: 100%; /* This could be ANY height */
    background: #ccc;
}

.inner-container {
    display: table-cell;
    vertical-align: middle;
    text-align: center;
}

.centered-content {
    display: inline-block;
    text-align: left;
    background: #fff;
    padding: 20px;
    border: 1px solid #000;
}
```

Voir aussi **ce violon** !

Centrage à taille fixe

Si la taille de votre contenu est fixe, vous pouvez utiliser le positionnement absolu à 50% avec une `margin` qui réduit de moitié la largeur et la hauteur de votre contenu:

HTML

```
<div class="center">
    Center vertically and horizontally
</div>
```

CSS

```
.center {
    position: absolute;
    background: #ccc;

    left: 50%;
    width: 150px;
    margin-left: -75px;  /* width * -0.5 */

    top: 50%;
    height: 200px;
    margin-top: -100px;  /* height * -0.5 */
}
```

Centrage horizontal avec seulement une largeur fixe

Vous pouvez centrer l'élément horizontalement même si vous ne connaissez pas la hauteur du contenu:

HTML

```
<div class="center">
    Center only horizontally
</div>
```

CSS

```
.center {
    position: absolute;
    background: #ccc;

    left: 50%;
    width: 150px;
    margin-left: -75px;  /* width * -0.5 */
}
```

Centrage vertical à hauteur fixe

Si vous connaissez la hauteur de l'élément, vous pouvez centrer l'élément verticalement:

HTML

```
<div class="center">
    Center only vertically
```

```
</div>
```

CSS

```
.center {
    position: absolute;
    background: #ccc;

    top: 50%;
    height: 200px;
    margin-top: -100px;  /* width * -0.5 */
}
```

En utilisant la marge: 0 auto;

Les objets peuvent être centrés en utilisant la `margin: 0 auto;` s'ils sont des éléments de bloc et ont une largeur définie.

HTML

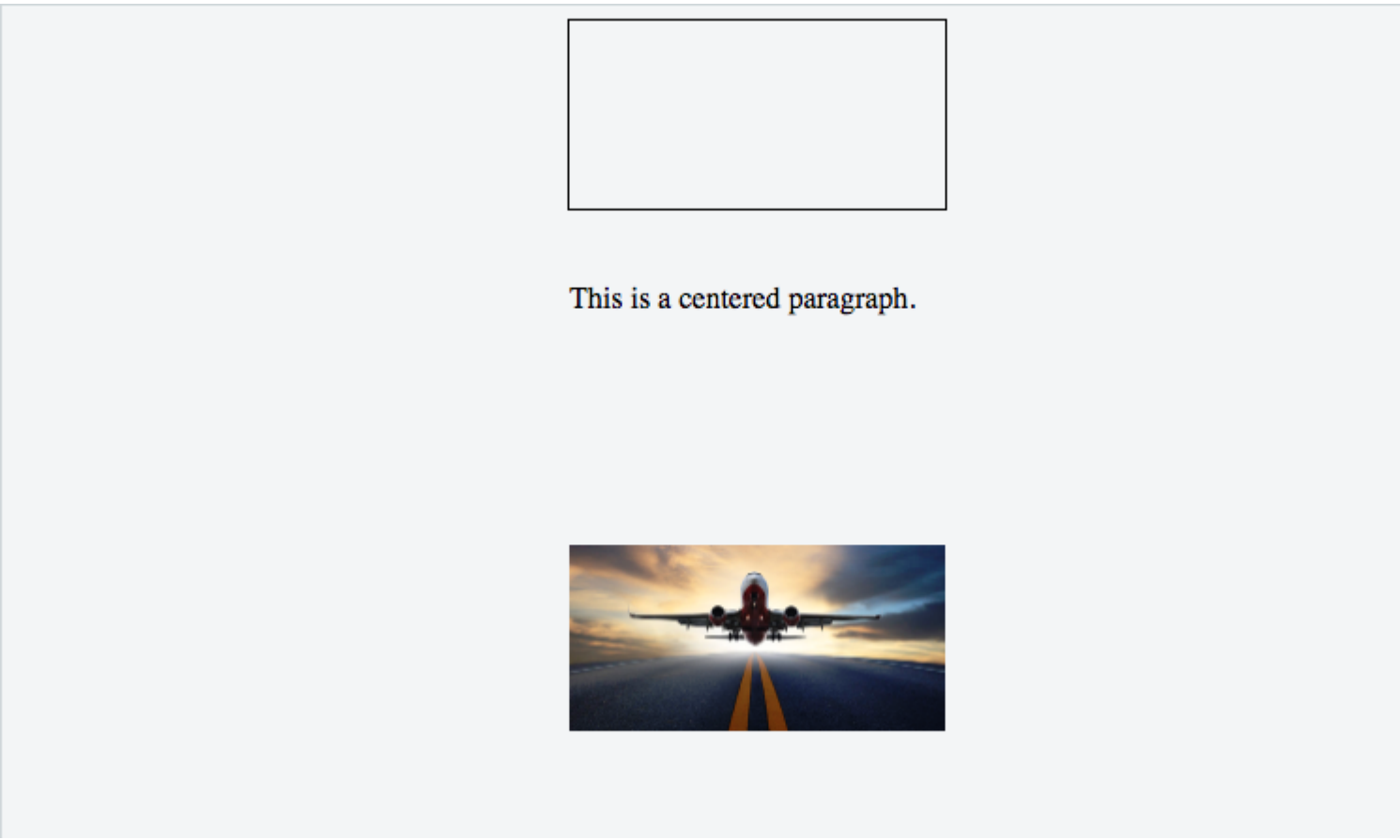
```
<div class="containerDiv">
    <div id="centeredDiv"></div>
</div>

<div class="containerDiv">
    <p id="centeredParagraph">This is a centered paragraph.</p>
</div>

<div class="containerDiv">
    <img id="centeredImage" src="https://i.kinja-img.com/gawker-media/image/upload/s--
c7Q9b4Eh--/c_scale,fl_progressive,q_80,w_800/qqyvc3bkpyl3mfhr8all.jpg" />
</div>
```

CSS

```
.containerDiv {
    width: 100%;
    height: 100px;
    padding-bottom: 40px;
}

#centeredDiv {
    margin: 0 auto;
    width: 200px;
    height: 100px;
    border: 1px solid #000;
}

#centeredParagraph {
    width: 200px;
    margin: 0 auto;
}

#centeredImage {
    display: block;
    width: 200px;
```

```
    margin: 0 auto;
}
```

Résultat:



JSFiddle example: Centrage des objets avec une marge: 0 auto;

Lire Centrage en ligne: https://riptutorial.com/fr/css/topic/299/centrage

Chapitre 7: Centrage vertical

Remarques

Ceci est utilisé lorsque les dimensions de l'élément (`width` et `height`) ne sont pas connues ou dynamiques.

Préférez utiliser **Flexbox** sur toutes les autres options, car il est optimisé pour la conception de l'interface utilisateur.

Examples

Centrage avec affichage: table

HTML:

```
<div class="wrapper">
    <div class="outer">
        <div class="inner">
            centered
        </div>
    </div>
</div>
```

CSS:

```
.wrapper {
  height: 600px;
  text-align: center;
}
.outer {
  display: table;
  height: 100%;
  width: 100%;
}
.outer .inner {
  display: table-cell;
  text-align: center;
  vertical-align: middle;
}
```

Centrage avec transformation

HTML:

```
<div class="wrapper">
    <div class="centered">
        centered
    </div>
</div>
```

CSS:

```
.wrapper {
  position: relative;
  height: 600px;
}
.centered {
  position: absolute;
  z-index: 999;
  transform: translate(-50%, -50%);
  top: 50%;
  left: 50%;
}
```

Centrage avec Flexbox

HTML:

```
<div class="container">
    <div class="child"></div>
</div>
```

CSS:

```
.container {
  height: 500px;
  width: 500px;
  display: flex;               // Use Flexbox
  align-items: center;         // This centers children vertically in the parent.
  justify-content: center;     // This centers children horizontally.
  background: white;
}

.child {
  width: 100px;
  height: 100px;
  background: blue;
}
```

Centrage du texte avec la hauteur de la ligne

HTML:

```
<div class="container">
    <span>vertically centered</span>
</div>
```

CSS:

```
.container{
    height: 50px;           /* set height */
    line-height: 50px;      /* set line-height equal to the height */
    vertical-align: middle; /* works without this rule, but it is good having it explicitly
set */
```

```
}
```

Remarque: cette méthode ne centre verticalement qu'une *seule ligne de texte* . Il ne centrera pas correctement les éléments de bloc et si le texte se divise en une nouvelle ligne, vous aurez deux grandes lignes de texte.

Centrage avec position: absolu

HTML:

```
<div class="wrapper">
  <img src="http://cdn.sstatic.net/Sites/stackoverflow/company/img/logos/so/so-
icon.png?v=c78bd457575a">
</div>
```

CSS:

```
.wrapper{
   position:relative;
   height: 600px;
}
.wrapper img {
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
  margin: auto;
}
```

Si vous voulez centrer d'autres images, vous devez donner de la hauteur et de la largeur à cet élément.

HTML:

```
<div class="wrapper">
  <div class="child">
     make me center
  </div>
</div>
```

CSS:

```
.wrapper{
   position:relative;
   height: 600px;
}
.wrapper .child {
  position: absolute;
  top: 0;
  left: 0;
  right: 0;
  bottom: 0;
```

```
  margin: auto;
  width: 200px;
  height: 30px;
  border: 1px solid #f00;
}
```

Centrage avec pseudo élément

HTML:

```
<div class="wrapper">
  <div class="content"></div>
</div>
```

CSS:

```
.wrapper{
   min-height: 600px;
}

.wrapper:before{
  content: "";
  display: inline-block;
  height: 100%;
  vertical-align: middle;
}

.content {
  display: inline-block;
  height: 80px;
  vertical-align: middle;
}
```

Cette méthode est mieux utilisée dans les cas où vous avez une hauteur variée `.content` centrée à l' intérieur `.wrapper` ; et que vous voulez `.wrapper` la hauteur de `.content` augmente lorsque la hauteur de `.wrapper` dépasse la hauteur `.wrapper` de `.wrapper` .

Lire Centrage vertical en ligne: https://riptutorial.com/fr/css/topic/5070/centrage-vertical

Chapitre 8: Colonnes

Syntaxe

- count_colonne: auto | nombre | inherit | initial | unset;
- column-width: auto | length;
- colonne: [colonne-largeur] | [colonne-nombre];
- span_colonne: none | all | inherit | initial | unset;
- gap de colonne: normal | length | inherit | initial | unset;
- fill-fill: auto | balance | inherit | intial | unset;
- column-rule-color: color | inherit | initial | unset;
- column-rule-style: none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset | inherit | initial | unset;
- column-rule-width: thin | medium | thick | length | inherit | initial | unset;
- règle-colonne: [column-rule-width] | [columm-rule-style] | [colonne-règle-couleur];
- break-after: auto | toujours | gauche | droite | recto | verso | page | colonne | région | éviter | éviter-page | éviter-colonne | éviter-région;
- break-before: auto | toujours | gauche | droite | recto | verso | page | colonne | région | éviter | éviter-page | éviter-colonne | éviter-région;
- break-inside: auto | détendre | éviter-page | éviter-colonne | éviter-région;

Examples

Exemple simple (nombre de colonnes)

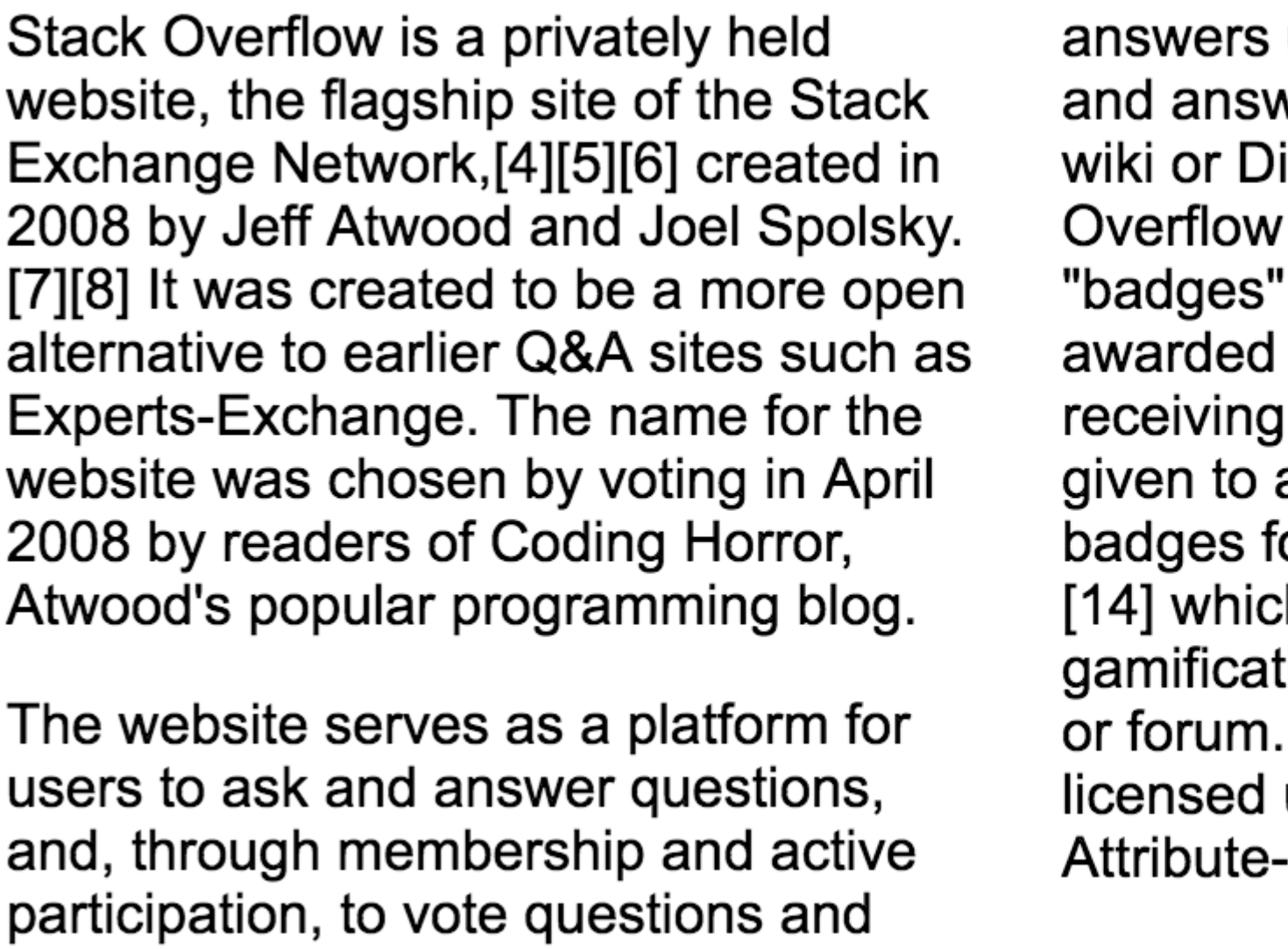
La disposition multi-colonnes CSS facilite la création de plusieurs colonnes de texte.

Code

```
<div id="multi-columns">Lorem ipsum 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</div>
```

```
.multi-columns {
  -moz-column-count: 2;
  -webkit-column-count: 2;
  column-count: 2;
}
```

Résultat



Largeur de colonne

La propriété `column-width` définit la largeur de colonne minimale. Si le nombre de `column-count` n'est pas défini, le navigateur affichera autant de colonnes que possible dans la largeur disponible.

Code:

```
<div id="multi-columns">
    Lorem ipsum 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
</div>
```

```
.multi-columns {
  -moz-column-width: 100px;
  -webkit-column-width: 100px;
  column-width: 100px;
}
```

Résultat

Stack Overflow is a privately held website, the flagship site of the Stack Exchange Network,[4][5] [6] created in 2008 by Jeff Atwood and Joel Spolsky. [7][8] It was created to be a more open alternative to earlier Q&A sites such as Experts-Exchange. The name for the website was chosen by voting in April 2008 by readers of Coding Horror, Atwood's popular programming blog.

The website serves as a platform for users to ask and answer questions, and, through membership and active participation, to vote questions and answers up or down and edit questions and answers in a fashion similar to a wiki or Digg.[13] Users of Stack Overflow can earn reputation points and "badges"; for example, a person is awarded 10 reputation points for receiving an "up" vote on an answer given to a question, and can receive badges for their valued contributions, [14] which represents a kind of gamification of the traditional Q&A site or forum. All user-generated content is licensed under a Creative Commons Attribute-ShareAlike license.

Lire Colonnes en ligne: https://riptutorial.com/fr/css/topic/3042/colonnes

Chapitre 9: commentaires

Syntaxe

- / * Commentaire * /

Remarques

- Les commentaires en CSS commencent toujours par `/*` et se terminent par `*/`
- Les commentaires ne peuvent pas être imbriqués

Examples

Une seule ligne

```
/* This is a CSS comment */
div {
   color: red; /* This is a CSS comment */
}
```

Ligne multiple

```
/*
    This
    is
    a
    CSS
    comment
*/
div {
   color: red;
}
```

Lire commentaires en ligne: https://riptutorial.com/fr/css/topic/1625/commentaires

Chapitre 10: Compteurs

Syntaxe

- Contre-ensemble: [<counter-name> <integer>?] + | aucun
- counter-reset: [<counter-name> <integer>?] + | aucun
- contre-incrémentation: [<counter-name> <integer>?] + | aucun
- counter (<counter-name> [, <counter-style>]?)
- counters (<counter-name>, <connector-string> [, <counter-style>]?)

Paramètres

Paramètre	Détails
nom du compteur	C'est le nom du compteur qui doit être créé, incrémenté ou imprimé. Il peut s'agir de n'importe quel nom personnalisé selon les souhaits du développeur.
entier	Cet entier est une valeur facultative qui, lorsqu'elle est fournie à côté du nom du compteur, représente la valeur initiale du compteur (dans le `counter-set` , les propriétés de `counter-reset`) ou la valeur à laquelle le compteur doit être incrémenté (en `counter-increment`).
aucun	C'est la valeur initiale pour les trois propriétés du `counter-*` . Lorsque cette valeur est utilisée pour un `counter-increment` de `counter-increment` , la valeur d'aucun des compteurs n'est affectée. Lorsque ceci est utilisé pour les deux autres, aucun compteur n'est créé.
contre-style	Ceci spécifie le style dans lequel la valeur du compteur doit être affichée. Il prend en charge toutes les valeurs prises en charge par la propriété `list-style-type` . Si `none` n'est utilisé, la valeur du compteur n'est pas imprimée du tout.
chaîne de connexion	Cela représente la chaîne qui doit être placée entre les valeurs de deux niveaux de compteur différents (comme le "." Dans "2.1.1").

Remarques

Les compteurs ne sont pas un nouveau sujet en CSS. Cela faisait partie des spécifications de niveau CSS 2 (révision 1 pour être précis) et a donc un support de navigateur très élevé.

Tous les navigateurs sauf IE6 et IE7 prennent en charge les compteurs CSS.

Examples

Application de chiffres romains à la sortie du compteur

CSS

```
body {
  counter-reset: item-counter;
}

.item {
  counter-increment: item-counter;
}

.item:before {
  content: counter(item-counter, upper-roman) ". "; /* by specifying the upper-roman as style
the output would be in roman numbers */
}
```

HTML

```
<div class='item'>Item No: 1</div>
<div class='item'>Item No: 2</div>
<div class='item'>Item No: 3</div>
```

Dans l'exemple ci-dessus, la sortie du compteur serait affichée en tant que I, II, III (chiffres romains) au lieu des 1, 2, 3 habituels, car le développeur a explicitement spécifié le style du compteur.

Numéroter chaque article en utilisant le compteur CSS

CSS

```
body {
  counter-reset: item-counter; /* create the counter */
}
.item {
  counter-increment: item-counter; /* increment the counter every time an element with class
"item" is encountered */
}
.item-header:before {
  content: counter(item-counter) ". "; /* print the value of the counter before the header and
append a "." to it */
}

/* just for demo */

.item {
```

```
  border: 1px solid;
  height: 100px;
  margin-bottom: 10px;
}
.item-header {
  border-bottom: 1px solid;
  height: 40px;
  line-height: 40px;
  padding: 5px;
}
.item-content {
  padding: 8px;
}
```

HTML

```
<div class='item'>
  <div class='item-header'>Item 1 Header</div>
  <div class='item-content'>Lorem Ipsum Dolor Sit Amet....</div>
</div>
<div class='item'>
  <div class='item-header'>Item 2 Header</div>
  <div class='item-content'>Lorem Ipsum Dolor Sit Amet....</div>
</div>
<div class='item'>
  <div class='item-header'>Item 3 Header</div>
  <div class='item-content'>Lorem Ipsum Dolor Sit Amet....</div>
</div>
```

L'exemple ci-dessus numérote chaque "élément" de la page et ajoute le numéro de l'élément avant son en-tête (en utilisant `content` propriété `content` de l'élément `.item-header :before` pseudo). Une démonstration en direct de ce code est disponible ici .

Implémentation de la numérotation multi-niveaux à l'aide des compteurs CSS

CSS

```
ul {
  list-style: none;
  counter-reset: list-item-number; /* self nesting counter as name is same for all levels */
}
li {
  counter-increment: list-item-number;
}
li:before {
  content: counters(list-item-number, ".") " "; /* usage of counters() function means value of
counters at all higher levels are combined before printing */
}
```

HTML

```
<ul>
  <li>Level 1
    <ul>
      <li>Level 1.1
        <ul>
          <li>Level 1.1.1</li>
        </ul>
      </li>
    </ul>
  </li>
  <li>Level 2
    <ul>
      <li>Level 2.1
        <ul>
          <li>Level 2.1.1</li>
          <li>Level 2.1.2</li>
        </ul>
      </li>
    </ul>
  </li>
  <li>Level 3</li>
</ul>
```

Ce qui précède est un exemple de numérotation à plusieurs niveaux utilisant des compteurs CSS. Il utilise le concept d' ***auto-imbrication*** des compteurs. L'imbrication automatique est un concept où si un élément a déjà un compteur avec le nom donné mais doit en créer un autre, il le crée en tant qu'enfant du compteur existant. Ici, le deuxième niveau `ul` hérite déjà du compteur `list-item-number` de son parent, mais doit ensuite créer son propre `list-item-number` (pour ses enfants `li`) et créer ainsi `list-item-number[1]` (counter for deuxième niveau) et le niche sous le `list-item-number[0]` (compteur pour le premier niveau). Ainsi, la numérotation à plusieurs niveaux est atteinte.

La sortie est imprimée à l'aide de la fonction `counters()` au lieu de la fonction `counter()` car la fonction `counters()` est conçue pour préfixer la valeur de tous les compteurs de niveau supérieur (parent) lors de l'impression de la sortie.

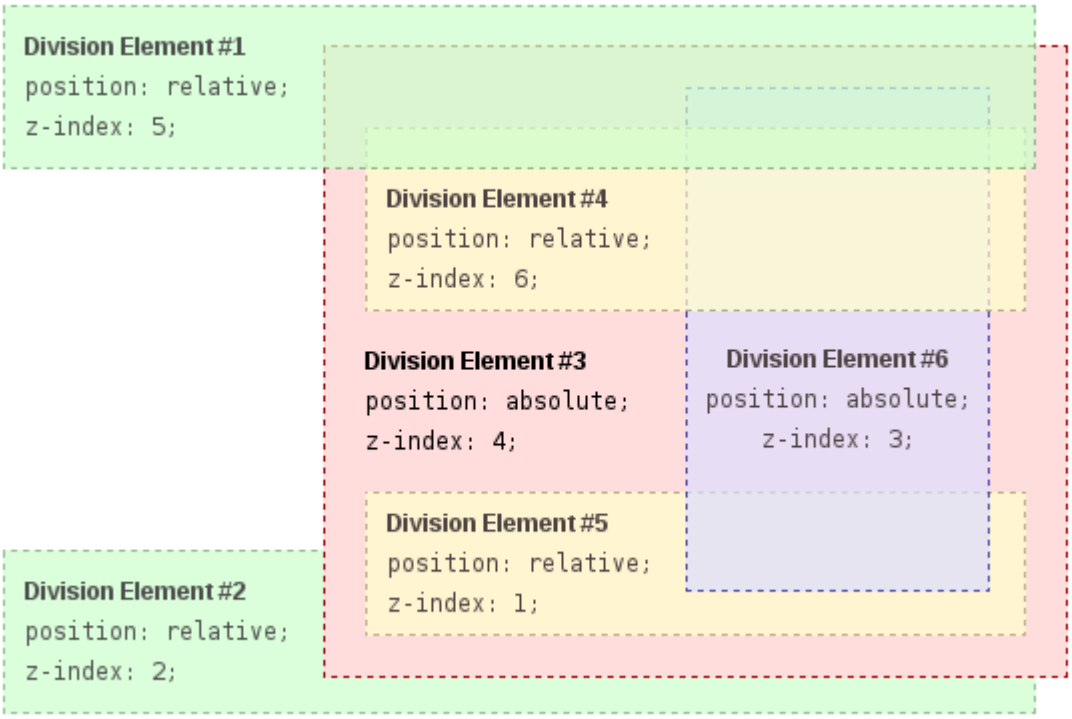
Lire Compteurs en ligne: https://riptutorial.com/fr/css/topic/2575/compteurs

Chapitre 11: Contexte d'empilement

Examples

Contexte d'empilement

Dans cet exemple, chaque élément positionné crée son propre contexte d'empilement, en raison de son positionnement et de ses valeurs d'index z. La hiérarchie des contextes d'empilement est organisée comme suit:



- Racine
  - DIV # 1
  - DIV # 2
  - DIV # 3
  - DIV # 4
  - DIV # 5
  - DIV # 6

Il est important de noter que DIV # 4, DIV # 5 et DIV # 6 sont des enfants de DIV # 3, de sorte que l'empilement de ces éléments est complètement résolu dans DIV # 3. Une fois que l'empilement et le rendu dans DIV # 3 sont terminés, tout l'élément DIV # 3 est transmis pour empilement dans l'élément racine par rapport à la DIV de son frère.

HTML:

```
<div id="div1">
  <h1>Division Element #1</h1>
```

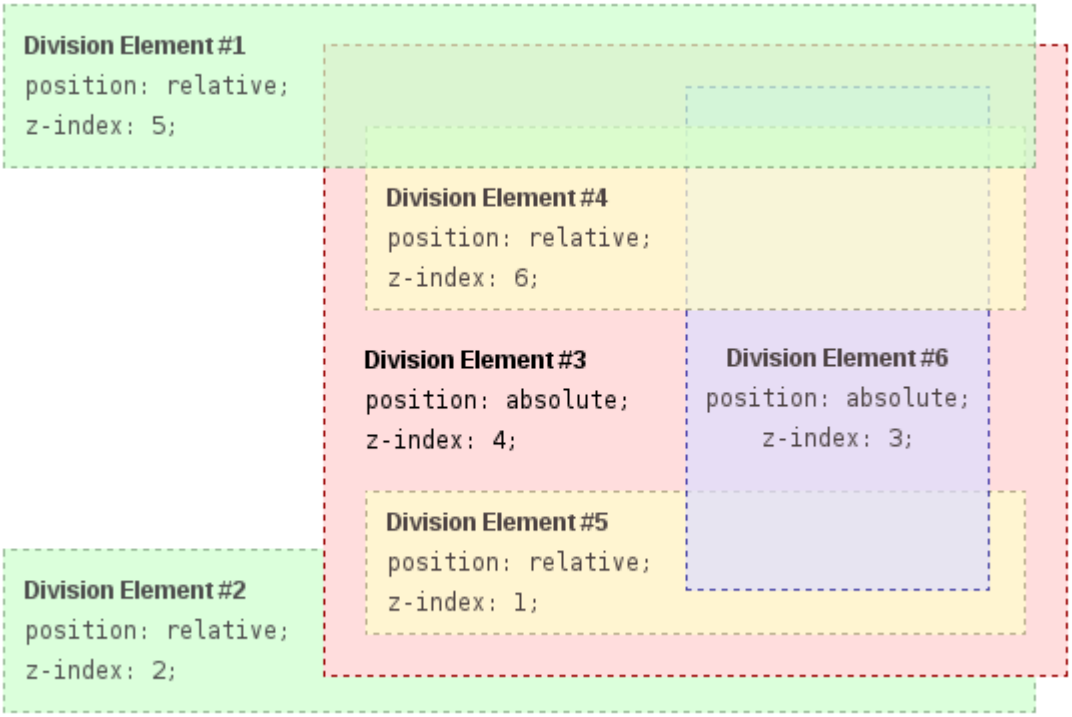
```
  <code>position: relative;<br/>
  z-index: 5;</code>
</div>
<div id="div2">
  <h1>Division Element #2</h1>
  <code>position: relative;<br/>
  z-index: 2;</code>
</div>
<div id="div3">
  <div id="div4">
    <h1>Division Element #4</h1>
    <code>position: relative;<br/>
    z-index: 6;</code>
  </div>
  <h1>Division Element #3</h1>
  <code>position: absolute;<br/>
  z-index: 4;</code>
  <div id="div5">
    <h1>Division Element #5</h1>
    <code>position: relative;<br/>
    z-index: 1;</code>
  </div>
  <div id="div6">
    <h1>Division Element #6</h1>
    <code>position: absolute;<br/>
    z-index: 3;</code>
  </div>
</div>
```

CSS:

```
* {
    margin: 0;
}
html {
    padding: 20px;
    font: 12px/20px Arial, sans-serif;
}
div {
    opacity: 0.7;
    position: relative;
}
h1 {
    font: inherit;
    font-weight: bold;
}
#div1,
#div2 {
    border: 1px dashed #696;
    padding: 10px;
    background-color: #cfc;
}
#div1 {
    z-index: 5;
    margin-bottom: 190px;
}
#div2 {
    z-index: 2;
}
#div3 {
```

```
    z-index: 4;
    opacity: 1;
    position: absolute;
    top: 40px;
    left: 180px;
    width: 330px;
    border: 1px dashed #900;
    background-color: #fdd;
    padding: 40px 20px 20px;
}
#div4,
#div5 {
    border: 1px dashed #996;
    background-color: #ffc;
}
#div4 {
    z-index: 6;
    margin-bottom: 15px;
    padding: 25px 10px 5px;
}
#div5 {
    z-index: 1;
    margin-top: 15px;
    padding: 5px 10px;
}
#div6 {
    z-index: 3;
    position: absolute;
    top: 20px;
    left: 180px;
    width: 150px;
    height: 125px;
    border: 1px dashed #009;
    padding-top: 125px;
    background-color: #ddf;
    text-align: center;
}
```

Résultat:



Source: https://developer.mozilla.org/en-US/docs/Web/CSS/CSS_Positioning/Understanding_z_index/The_stacking_context .

Lire Contexte d'empilement en ligne: https://riptutorial.com/fr/css/topic/5037/contexte-d-empilement

Chapitre 12: Contextes de mise en forme de bloc

Remarques

> [Un contexte de formatage de bloc fait partie d'un rendu CSS visuel d'une page Web. C'est la région dans laquelle la disposition des blocs se produit et dans laquelle les flotteurs interagissent les uns avec les autres.] [1]

[1]: https://developer.mozilla.org/en-US/docs/Web/Guide/CSS/Block_formatting_context MDN

Examples

Utilisation de la propriété overflow avec une valeur différente de visible

```
img{
  float:left;
  width:100px;
  margin:0 10px;
}
.div1{
  background:#f1f1f1;
  /* does not create block formatting context */
}
.div2{
  background:#f1f1f1;
  overflow:hidden;
  /* creates block formatting context */
}
```

ate Fork Set as base Tidy Collaborate Embed

HTML

```
<img src="https://static.pexels.com/photos/33695/matryoshka-red-fly-agaric-mushroom-mushrooms-forest-medium.jpg" alt="" />
<div class=div1>
<p>Lorem ipsum dolor sit amet, cum no paulo mollis pertinacia. Eam in velit graecis, sea mucius insolens ne. Amet doming at has, omnis errem an cum. Eu vim appareat persecuti, ea putant definitionem has, vis ea legendos expetenda. No eros graeci minimum nam, justo augue instructior usu ne. At ludus suscipit disputationi vel.</p>

<p>Ad case omnis nam, mutat deseruisse persequeris eos ad, in tollit debitis sea. Cu eos munere virtute vituperata. Exerci bonorum sed id, id nec tantas praesent complectitur. Vel cu legendos mediocritatem. Enim liberavisse ei sea.</p>
</div>

<img src="https://static.pexels.com/photos/129574/pexels-photo-129574-medium.jpeg" alt="" />
<div class=div2>
<p>Lorem ipsum dolor sit amet, cum no paulo mollis pertinacia. Eam in velit graecis, sea mucius insolens ne. Amet doming at has, omnis errem an cum. Eu vim appareat persecuti, ea putant definitionem has, vis ea legendos expetenda. No eros graeci minimum nam, justo augue instructior usu ne. At ludus suscipit disputationi vel.</p>

<p>Ad case omnis nam, mutat deseruisse persequeris eos ad, in tollit debitis sea. Cu eos munere virtute vituperata. Exerci bonorum sed id, id nec tantas praesent complectitur. Vel cu legendos mediocritatem. Enim liberavisse ei sea.</p>
</div>
```

```
img{
  float:left;
  width:100px;
  margin:0 10px;
}
.div1{
  background:#f1f1f1;
}
.div2{
  background:#f1f1f1;
  overflow:hidden;
  /* creates block formatting c
}
```

Lorem ipsum dolor
doming at has, omni
expetenda. No eros

Ad case omnis nam, mutat deseruisse
sed id, id nec tantas praesent complecti

Lorem ipsum dolor
doming at has, omni
expetenda. No eros

Ad case omnis nam,
Exerci bonorum sed
sea.

https://jsfiddle.net/MadalinaTn/qkwwmu6m/2/

> L'utilisation de la propriété overflow avec une valeur différente de visible (sa valeur par défaut) créera un nouveau contexte de mise en forme de bloc. C'est techniquement nécessaire - si un flotteur intersecté avec l'élément de défilement, il ré-enroulerait de force le contenu.

Cet exemple qui montre comment un certain nombre de paragraphes interagiront avec une image flottante est similaire à cet exemple , sur css-tricks.com.

2 : https://developer.mozilla.org/en-US/docs/Web/CSS/overflow MDN

Lire Contextes de mise en forme de bloc en ligne:
https://riptutorial.com/fr/css/topic/5069/contextes-de-mise-en-forme-de-bloc

Chapitre 13: Contrôle de disposition

Syntaxe

- affichage: aucun | inline | bloc | item de liste | inline-list-item | bloc en ligne | table en ligne | table | cellule de table | colonne de table | table-colonne-groupe | groupe de pieds de table | table-en-tête-groupe | rangée de table | table-rangée-groupe | flex | inline-flex | grille | grille en ligne | rodage | rubis | base de rubis | ruby-text | ruby-base-container | ruby-text-container | Contenu;

Paramètres

Valeur	Effet
`none`	Cacher l'élément et l'empêcher d'occuper l'espace.
`block`	Elément de bloc, occuper 100% de la largeur disponible, rupture après élément.
`inline`	Elément en ligne, n'occupe aucune largeur, aucun élément après rupture.
`inline-block`	Prendre des propriétés spéciales à la fois des éléments en ligne et des éléments de bloc, pas de rupture, mais peut avoir une largeur.
`inline-flex`	Affiche un élément en tant que conteneur flexible de niveau intégré.
`inline-table`	L'élément est affiché sous la forme d'une table de niveau en ligne.
`grid`	Se comporte comme un élément de bloc et affiche son contenu en fonction du modèle de grille.
`flex`	Se comporte comme un élément de bloc et affiche son contenu selon le modèle Flexbox.
`inherit`	Héritez la valeur de l'élément parent.
`initial`	Rétablissez la valeur par défaut à partir des comportements décrits dans les spécifications HTML ou à partir de la feuille de style par défaut du navigateur / utilisateur.
`table`	Se comporte comme l'élément de `table` HTML.
`table-cell`	Laisser l'élément se comporter comme un élément `<td>`
`table-column`	Laisser l'élément se comporter comme un élément `<col>`

Valeur	Effet
`table-row`	Laisser l'élément se comporter comme un élément `<tr>`
`list-item`	Laisser l'élément se comporter comme un élément `<li>` .

Examples

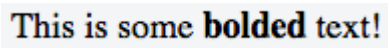
La propriété d'affichage

La propriété CSS d' `display` est fondamentale pour contrôler la disposition et le flux d'un document HTML. La plupart des éléments ont une valeur d' `display` par défaut de type `block` ou `inline` (bien que certains éléments aient d'autres valeurs par défaut).

En ligne

Un élément en `inline` n'occupe que la largeur nécessaire. Il empile horizontalement avec d'autres éléments du même type et peut ne pas contenir d'autres éléments non intégrés.

```
<span>This is some <b>bolded</b> text!</span>
```

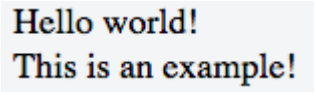


Comme démontré ci-dessus, deux éléments en `inline` , `<span>` et `<b>` , sont en ligne (d'où le nom) et ne rompent pas le flux du texte.

Bloc

Un élément de `block` occupe la largeur maximale disponible de son élément parent. Il commence par une nouvelle ligne et, contrairement aux éléments en `inline` , ne restreint pas le type d'éléments qu'il peut contenir.

```
<div>Hello world!</div><div>This is an example!</div>
```



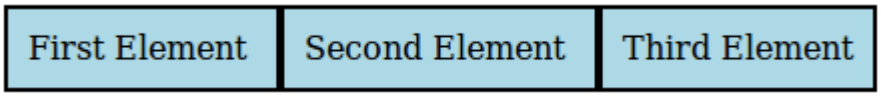
L'élément `div` est au niveau du bloc par défaut et, comme indiqué ci-dessus, les deux éléments de `block` sont empilés verticalement et, contrairement aux éléments en `inline` , le flux du texte est interrompu.

Bloc Inline

La valeur `inline-block` nous donne le meilleur des deux mondes: elle mélange l'élément avec le flux du texte tout en nous permettant d'utiliser des propriétés de `padding` , de `margin` , de `height` et similaires qui n'ont aucun effet visible sur `inline` éléments en `inline` .

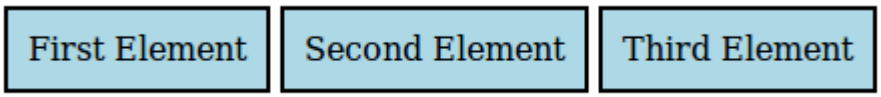
Les éléments avec cette valeur d'affichage agissent comme s'ils étaient du texte normal et, par conséquent, ils sont affectés par les règles contrôlant le flux de texte, telles que `text-align` . Par défaut, ils sont réduits à la plus petite taille possible pour accueillir leur contenu.

```
<!--Inline: unordered list-->
<style>
li {
    display : inline;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }
</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```




```
<!--block: unordered list-->
<style>
li {
    display : block;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }
</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```


```
<!--Inline-block: unordered list-->
<style>
li {
    display : inline-block;
    background : lightblue;
    padding:10px;

    border-width:2px;
    border-color:black;
    border-style:solid;
    }

</style>

<ul>
<li>First Element </li>
<li>Second Element </li>
<li>Third Element </li>
</ul>
```



aucun

Un élément auquel est donnée la valeur none à sa propriété d'affichage ne sera pas affiché du tout.

Par exemple, créons un élément div avec un identifiant de `myDiv` :

```
<div id="myDiv"></div>
```

Cela peut maintenant être marqué comme n'étant pas affiché par la règle CSS suivante:

```
#myDiv {
    display: none;
}
```

Lorsqu'un élément a été défini pour être `display:none;` le navigateur ignore toutes les autres propriétés de mise en page pour cet élément spécifique (à la fois la `position` et le `float`). Aucune case ne sera rendue pour cet élément et son existence en HTML n’affecte pas la position des éléments suivants.

Notez que cela diffère de la définition de la propriété de `visibility` sur `hidden` . Réglage de la `visibility: hidden;` car un élément n'afficherait pas l'élément sur la page mais l'élément occuperait toujours l'espace dans le processus de rendu comme s'il était visible. Cela affectera donc la façon dont les éléments suivants sont affichés sur la page.

La valeur `none` pour la propriété display est généralement utilisée avec JavaScript pour afficher ou masquer des éléments à volonté, ce qui élimine le besoin de les supprimer et de les recréer.

Pour obtenir l'ancienne structure de table en utilisant div

Ceci est la structure de table HTML normale

```
<style>
    table {
        width: 100%;
    }
</style>

<table>
  <tr>
    <td>
        I'm a table
    </td>
  </tr>
</table>
```

Vous pouvez faire la même implémentation comme celle-ci

```
<style>
    .table-div {
        display: table;
    }
    .table-row-div {
        display: table-row;
    }
    .table-cell-div {
        display: table-cell;
    }
</style>

<div class="table-div>
  <div class="table-row-div>
    <div class="table-cell-div>
      I behave like a table now
    </div>
  </div>
</div>
```

Lire Contrôle de disposition en ligne: https://riptutorial.com/fr/css/topic/1473/controle-de-disposition

Chapitre 14: Couleurs

Syntaxe

- couleur: #rgb
- couleur: #rrggbb
- couleur: rgb [a] (<red>, <green>, <blue> [, <alpha>])
- couleur: hsl [a] (<hue>, <saturation%>, <lightness%> [, <alpha>])
- couleur: colorkeyword / * vert, bleu, jaune, orange, rouge, ..etc * /

Examples

Mots-clés de couleur

La plupart des navigateurs utilisent des mots-clés en couleur pour spécifier une couleur. Par exemple, pour définir la `color` d'un élément sur bleu, utilisez le mot-clé `blue` :

```
.some-class {
    color: blue;
}
```

Mots - clés CSS ne sont pas les majuscules sensitive- `blue` , `Blue` et `BLUE` generera dans `#0000FF` .

Mots-clés de couleur

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
AliceBlue	# F0F8FF	rgb (240,248,255)	
Blanc antique	# FAEBD7	rgb (250,235,215)	
Aqua	# 00FFFF	rgb (0,255,255)	
Bleu vert	# 7FFFD4	rgb (127,255,212)	
Azur	# F0FFFF	rgb (240 255 255)	
Beige	# F5F5DC	rgb (245,245,220)	
Bisque	# FFE4C4	rgb (255.228.196)	
Noir	# 000000	rgb (0,0,0)	

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
BlanchiAlmond	#FFEBCD	rgb (255,235,205)	
Bleu	# 0000FF	rgb (0,0,255)	
BlueViolet	# 8A2BE2	rgb (138,43,226)	
marron	# A52A2A	rgb (165,42,42)	
Bois bohu	# DEB887	rgb (222,184,135)	
CadetBlue	# 5F9EA0	rgb (95,158,160)	
Chartreuse	# 7FFF00	rgb (127 255,0)	
Chocolat	# D2691E	rgb (210,105,30)	
corail	# FF7F50	rgb (255,127,80)	
Bleuet	# 6495ED	rgb (100 149 237)	
Soie de maïs	# FFF8DC	rgb (255 248 220)	
cramoisi	# DC143C	rgb (220,20,60)	
Cyan	# 00FFFF	rgb (0,255,255)	
Bleu foncé	# 00008B	rgb (0,0,139)	
DarkCyan	# 008B8B	rgb (0,139,139)	
DarkGoldenRod	# B8860B	rgb (184,134,11)	
Gris foncé	# A9A9A9	rgb (169,169,169)	
Gris foncé	# A9A9A9	rgb (169,169,169)	
Vert foncé	# 006400	rgb (0,100,0)	
DarkKhaki	# BDB76B	rgb (189,183,107)	
DarkMagenta	# 8B008B	rgb (139,0,139)	

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
DarkOliveGreen	# 556B2F	rgb (85,107,47)	
Orange sombre	# FF8C00	rgb (255,140,0)	
DarkOrchid	# 9932CC	rgb (153,50,204)	
Rouge foncé	# 8B0000	rgb (139,0,0)	
DarkSalmon	# E9967A	rgb (233,150,122)	
DarkSeaGreen	# 8FBC8F	RVB (143 188 143)	
DarkSlateBlue	# 483D8B	rgb (72,61,139)	
DarkSlateGray	# 2F4F4F	rgb (47,79,79)	
DarkSlateGrey	# 2F4F4F	rgb (47,79,79)	
DarkTurquoise	# 00CED1	rgb (0,206,209)	
DarkViolet	# 9400D3	rgb (148,0,211)	
Rose profond	# FF1493	rgb (255,20,147)	
DeepSkyBlue	# 00BFFF	rgb (0,191,255)	
DimGray	# 696969	rgb (105,105,105)	
DimGrey	# 696969	rgb (105,105,105)	
DodgerBlue	# 1E90FF	rgb (30,144,255)	
FireBrick	# B22222	rgb (178,34,34)	
FloralWhite	# FFFAF0	rgb (255 250 240)	
Forêt verte	# 228B22	rgb (34,139,34)	
Fuchsia	# FF00FF	rgb (255,0,255)	
Gainsboro	#DCDCDC	rgb (220 220 220)	

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
GhostWhite	# F8F8FF	rgb (248,248,255)	
Or	# FFD700	rgb (255,215,0)	
GoldenRod	# DAA520	rgb (218,165,32)	
Gris	# 808080	rgb (128,128,128)	
Gris	# 808080	rgb (128,128,128)	
vert	# 008000	rgb (0,128,0)	
Vert jaune	# ADFF2F	rgb (173 255,47)	
Miellat	# F0FFF0	rgb (240 255 240)	
Rose vif	# FF69B4	rgb (255,105,180)	
IndianRed	# CD5C5C	rgb (205,92,92)	
Indigo	# 4B0082	rgb (75,0,130)	
Ivoire	# FFFFF0	rgb (255,255,240)	
Kaki	# F0E68C	rgb (240,230,140)	
Lavande	# E6E6FA	rgb (230 230 250)	
LavandeBlush	# FFF0F5	rgb (255,240,245)	
LawnGreen	# 7CFC00	rgb (124,252,0)	
Citronchiffon	#FFFACD	rgb (255 250 205)	
Bleu clair	# ADD8E6	rgb (173,216,230)	
LightCoral	# F08080	rgb (240,128,128)	
Cyan clair	# E0FFFF	rgb (224,255,255)	
LightGoldenRodYellow	# FAFAD2	rgb (250 250 210)	

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
Gris clair	# D3D3D3	rgb (211,211,211)	
Gris clair	# D3D3D3	rgb (211,211,211)	
Vert clair	# 90EE90	rgb (144.238.144)	
Rose clair	# FFB6C1	rgb (255,182,193)	
LightSalmon	# FFA07A	rgb (255,160,122)	
LightSeaGreen	# 20B2AA	rgb (32,178,170)	
LightSkyBlue	# 87CEFA	rgb (135 206 250)	
LightSlateGray	# 778899	rgb (119,136,153)	
LightSlateGrey	# 778899	rgb (119,136,153)	
LightSteelBlue	# B0C4DE	rgb (176,196,222)	
Jaune clair	# FFFFE0	rgb (255,255,224)	
Citron vert	# 00FF00	rgb (0,255,0)	
Vert citron	# 32CD32	rgb (50,205,50)	
Lin	# FAF0E6	rgb (250,240,230)	
Magenta	# FF00FF	rgb (255,0,255)	
Bordeaux	# 800000	rgb (128,0,0)	
MoyenAquaMarine	# 66CDAA	rgb (102,205,170)	
MediumBlue	# 0000CD	rgb (0,0,205)	
MediumOrchid	# BA55D3	rgb (186,85,211)	
MediumPurple	# 9370DB	rgb (147,112,219)	
MediumSeaGreen	# 3CB371	rgb (60,179,113)	

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
MediumSlateBlue	# 7B68EE	rgb (123,104,238)	
MediumSpringGreen	# 00FA9A	rgb (0,250,154)	
MediumTurquoise	# 48D1CC	rgb (72,209,204)	
MediumVioletRed	# C71585	rgb (199,21,133)	
Bleu nuit	# 191970	rgb (25,25,112)	
MintCream	# F5FFFA	rgb (245 255 250)	
MistyRose	# FFE4E1	rgb (255,228,225)	
Mocassin	# FFE4B5	rgb (255.228.181)	
Navajo Blanc	#FFDEAD	rgb (255,222,173)	
Marine	# 000080	rgb (0,0,128)	
OldLace	# FDF5E6	rgb (253,245,230)	
olive	# 808000	rgb (128,128,0)	
OliveDrab	# 6B8E23	rgb (107 142,35)	
Orange	# FFA500	rgb (255,165,0)	
Rouge-orange	# FF4500	rgb (255,69,0)	
Orchidée	# DA70D6	rgb (218,112,214)	
PaleGoldenRod	# EEE8AA	rgb (238,232,170)	
Vert pâle	# 98FB98	rgb (152,251,152)	
PaleTurquoise	#AFEEEE	rgb (175,238,238)	
PaleVioletRed	# DB7093	rgb (219,112,147)	
PapayaWhip	# FFEFD5	rgb (255,239,213)	

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
PeachPuff	# FFDAB9	rgb (255 218 185)	
Pérou	# CD853F	rgb (205,133,63)	
Rose	# FFC0CB	rgb (255,192,203)	
prune	# DDA0DD	rgb (221,160,221)	
Poudre bleue	# B0E0E6	rgb (176,224,230)	
Violet	# 800080	rgb (128,0,128)	
RebeccaPurple	# 663399	rgb (102,51,153)	
rouge	# FF0000	rgb (255,0,0)	
RosyBrown	# BC8F8F	RVB (188 143 143)	
Bleu royal	# 4169E1	rgb (65,105,225)	
SaddleBrown	# 8B4513	rgb (139,69,19)	
Saumon	# FA8072	rgb (250,128,114)	
SandyBrown	# F4A460	rgb (244,164,96)	
SeaGreen	# 2E8B57	rgb (46,139,87)	
SeaShell	# FFF5EE	rgb (255,245,238)	
Terre de sienne	# A0522D	rgb (160,82,45)	
argent	# C0C0C0	rgb (192,192,192)	
Bleu ciel	# 87CEEB	rgb (135 206 235)	
SlateBlue	# 6A5ACD	rgb (106,90,205)	
Gris ardoise	# 708090	rgb (112,128,144)	
Gris ardoise	# 708090	rgb (112,128,144)	

Nom de la couleur	Valeur hexadécimale	Valeurs RVB	Couleur
Neige	#FFFAFA	rgb (255 250 250)	
Vert printanier	# 00FF7F	rgb (0,255,127)	
SteelBlue	# 4682B4	rgb (70 130 180)	
bronzer	# D2B48C	RVB (210,180,140)	
Sarcelle	# 008080	rgb (0,128,128)	
Chardon	# D8BFD8	rgb (216,191,216)	
Tomate	# FF6347	rgb (255,99,71)	
Turquoise	# 40E0D0	rgb (64,224,208)	
Violet	# EE82EE	rgb (238,130,238)	
Blé	# F5DEB3	rgb (245,222,179)	
blanc	#FFFFFF	rgb (255,255,255)	
Fumée blanche	# F5F5F5	rgb (245,245,245)	
Jaune	# FFFF00	rgb (255,255,0)	
Vert jaunâtre	# 9ACD32	rgb (154,205,50)	

En plus des couleurs nommées, il y a aussi le mot-clé `transparent` , qui représente un noir entièrement transparent: `rgba(0,0,0,0)`

Valeur hexadécimale

Contexte

Les couleurs CSS peuvent également être représentées sous la forme d'un triplet hexadécimal, où les membres représentent les composants rouge, vert et bleu d'une couleur. Chacune de ces valeurs représente un nombre compris entre `00` et `FF` , ou `0` à `255` en notation décimale. Des valeurs Hexidécimales majuscules et / ou minuscules peuvent être utilisées (par exemple, `#3fc` = `#3FC` = `#33ffCC`). Le navigateur interprète `#369` comme `#336699` . Si ce n'est pas ce que vous vouliez,

mais plutôt le `#306090` , vous devez le spécifier explicitement.

Le nombre total de couleurs pouvant être représentées avec une notation hexadécimale est 256 ^ 3 ou 16 777 216.

Syntaxe

```
color: #rrggbb;
color: #rgb
```

Valeur	La description
`rr`	`00` - `FF` pour la quantité de rouge
`gg`	`00` - `FF` pour la quantité de vert
`bb`	`00` - `FF` pour la quantité de bleu

```
.some-class {
    /* This is equivalent to using the color keyword 'blue' */
    color: #0000FF;
}

.also-blue {
    /* If you want to specify each range value with a single number, you can!
       This is equivalent to '#0000FF' (and 'blue') */
    color: #00F;
}
```

La notation hexadécimale est utilisée pour spécifier les valeurs de couleur dans le format de couleur RVB, conformément aux «valeurs de couleur numériques» du W3C .

De nombreux outils sont disponibles sur Internet pour rechercher des valeurs de couleur hexadécimales (ou simplement hexadécimales).

Recherchez " **palette de couleurs hexadécimales** " ou " **sélecteur de couleurs hexadécimal** " avec votre navigateur Web préféré pour trouver un tas d'options!

Les valeurs hexadécimales commencent toujours par un signe dièse (#), durent jusqu'à six "chiffres" et ne sont pas sensibles à la casse: elles ne se soucient pas de la capitalisation. `#FFC125` et `#ffc125` sont de la même couleur.

rgb () Notation

RVB est un modèle de couleur additif qui représente les couleurs sous forme de mélanges de lumière rouge, verte et bleue. Essentiellement, la représentation RVB est l'équivalent décimal de la notation hexadécimale. En hexadécimal, chaque nombre est compris entre 00-FF, ce qui équivaut à 0-255 en décimal et 0% à 100% en pourcentage.

```
.some-class {
    /* Scalar RGB, equivalent to 'blue'*/
    color: rgb(0, 0, 255);
}

.also-blue {
    /* Percentile RGB values*/
    color: rgb(0%, 0%, 100%);
}
```

Syntaxe

```
rgb(<red>, <green>, <blue>)
```

Valeur	La description
`<red>`	un entier de 0 à 255 ou un pourcentage de 0 à 100%
`<green>`	un entier de 0 à 255 ou un pourcentage de 0 à 100%
`<blue>`	un entier de 0 à 255 ou un pourcentage de 0 à 100%

hsl () Notation

HSL signifie **teinte** ("quelle couleur"), **saturation** ("combien de couleur") et **légèreté** ("combien de blanc").

La teinte est représentée par un angle de 0 ° à 360 ° (sans unités), tandis que la saturation et la légèreté sont représentées sous forme de pourcentages.

```
p {
    color: hsl(240, 100%, 50%); /* Blue */
}
```

Syntaxe

```
color: hsl(<hue>, <saturation>%, <lightness>%);
```

Valeur	La description
`<hue>`	spécifié en degrés autour de la roue chromatique (sans unités), où 0 ° est rouge, 60 ° est jaune, 120 ° est vert, 180 ° est cyan, 240 ° est bleu, 300 ° est magenta et 360 ° est rouge
`<saturation>`	spécifié en pourcentage où 0% est complètement désaturé (niveaux de gris) et 100% est complètement saturé (couleur vive)

Valeur	La description
`<lightness>`	spécifié en pourcentage où 0% est entièrement noir et 100% entièrement blanc

Remarques

- Une saturation de 0% produit toujours une couleur en niveaux de gris; changer la teinte n'a aucun effet.
- Une légèreté de 0% produit toujours du noir et 100% produit toujours du blanc; changer la teinte ou la saturation n'a aucun effet.

currentColor

`currentColor` renvoie la valeur de couleur calculée de l'élément en cours.

Utiliser dans le même élément

Ici, la propriété currentColor est définie sur `red` car la propriété de `color` est définie sur `red` :

```
div {
   color: red;
   border: 5px solid currentColor;
   box-shadow: 0 0 5px currentColor;
}
```

Dans ce cas, la spécification de currentColor pour la bordure est probablement redondante car son omission devrait produire des résultats identiques. Utilisez uniquement la propriété currentColor à l'intérieur de la propriété border dans le même élément si elle est remplacée par un sélecteur plus spécifique .

Étant donné que c'est la couleur calculée, la bordure sera verte dans l'exemple suivant en raison de la seconde règle remplaçant le premier:

```
div {
   color: blue;
   border: 3px solid currentColor;
   color: green;
}
```

Hérité de l'élément parent

La couleur du parent est héritée, ici currentColor est évalué à 'bleu', ce qui rend la couleur de bordure de l'élément enfant bleue.

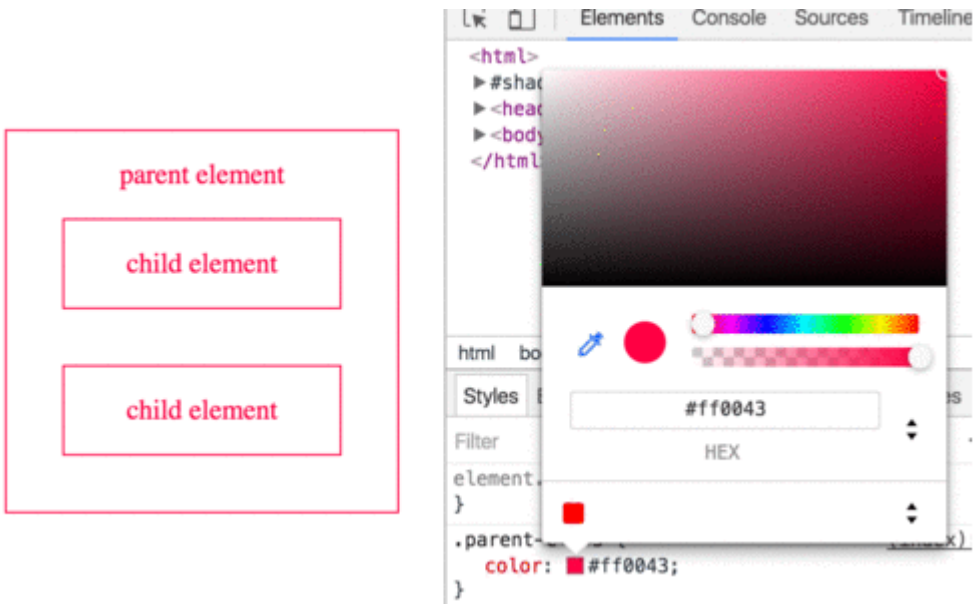
```
.parent-class {
```

```
    color: blue;
}

.parent-class .child-class {
    border-color: currentColor;
}
```

currentColor peut également être utilisé par d'autres règles qui, normalement, n'hériteraient pas de la propriété color, comme la couleur d'arrière-plan. L'exemple ci-dessous montre les enfants utilisant la couleur définie dans le parent comme arrière-plan:

```
.parent-class {
    color: blue;
}

.parent-class .child-class {
    background-color: currentColor;
}
```

Résultat possible:



rgba () Notation

Similaire à la notation rgb () , mais avec une valeur alpha (opacité) supplémentaire.

```
.red {
    /* Opaque red */
    color: rgba(255, 0, 0, 1);
}

.red-50p {
    /* Half-translucent red. */
    color: rgba(255, 0, 0, .5);
}
```

Syntaxe

```
rgba(<red>, <green>, <blue>, <alpha>);
```

Valeur	La description
`<red>`	un entier de 0 à 255 ou un pourcentage de 0 à 100%
`<green>`	un entier de 0 à 255 ou un pourcentage de 0 à 100%
`<blue>`	un entier de 0 à 255 ou un pourcentage de 0 à 100%
`<alpha>`	un nombre compris entre 0 et 1, où 0.0 est totalement transparent et 1.0 totalement opaque

hsla () Notation

Similaire à la notation hsl () , mais avec une valeur alpha (opacité) ajoutée.

```
hsla(240, 100%, 50%, 0)     /* transparent */
hsla(240, 100%, 50%, 0.5)   /* half-translucent blue */
hsla(240, 100%, 50%, 1)     /* fully opaque blue */
```

Syntaxe

```
hsla(<hue>, <saturation>%, <lightness>%, <alpha>);
```

Valeur	La description
`<hue>`	spécifié en degrés autour de la roue chromatique (sans unités), où 0 ° est rouge, 60 ° est jaune, 120 ° est vert, 180 ° est cyan, 240 ° est bleu, 300 ° est magenta et 360 ° est rouge
`<saturation>`	pourcentage où 0% est complètement désaturé (échelle de gris) et 100% est complètement saturé (couleur vive)
`<lightness>`	pourcentage où 0% est entièrement noir et 100% entièrement blanc
`<alpha>`	un nombre de 0 à 1 où 0 est totalement transparent et 1 est complètement opaque

Lire Couleurs en ligne: https://riptutorial.com/fr/css/topic/644/couleurs

Chapitre 15: Couper et Masquer

Syntaxe

- **Coupure**
- chemin de clip: <source-clip> | [<basic-shape> || <boîte-géométrie-clip>] | aucun
- **Masquage**
- image-masque: [aucun | <référence du masque>] #
- mode masque: [<masque-mode>] #
- mask-repeat: [<style de répétition] #
- position du masque: [<position>] #
- masque-clip: [<boîte-de-géométrie> | no-clip] #
- mask-origin: [<boîte-de-géométrie>] #
- taille de masque: [<bg-size>] #
- mask-composite: [<compositing-operator>] #
- mask: [<masque-référence> <mode masquage>? || <position> [/ <bg-size>]? || <style de répétition> || <boîte de géométrie> || [<boîte à géométrie> | no-clip] || <opérateur de composition>] #

Paramètres

Paramètre	Détails
source de clip	Une URL qui peut pointer vers un élément SVG incorporé (ou) un élément SVG dans un fichier externe contenant la définition du chemin du clip.
forme de base	Fait référence à un `inset()` parmi `inset()` , `circle()` , `ellipse()` ou `polygon()` . En utilisant l'une de ces fonctions, le chemin de détourage est défini. Ces fonctions de forme fonctionnent exactement de la même manière que dans les formes pour les flotteurs
boîte à géométrie de clip	Cela peut avoir une valeur parmi `content-box` , `padding-box` , `border-box` , `margin-box` , `fill-box` , `stroke-box` , `stroke-box view-box` . Lorsque ceci est fourni sans aucune valeur pour <basic-shape>, les bords de la boîte correspondante sont utilisés comme chemin pour le découpage. Utilisé avec un <basic-shape>, il sert de boîte de référence pour la forme.
référence de masque	Cela peut être `none` ou une image ou une URL de référence à une source d'image de masque.
style répétitif	Ceci spécifie comment le masque doit être répété ou classé dans les axes X et Y. Les valeurs prises en charge sont la `repeat-x` , la `repeat-y` , la `repeat` , l' `space` , le `round` , la `no-repeat` .
mode	Peut être `alpha` ou `luminance` ou `auto` et indique si le masque doit être traité

Paramètre	Détails
masque	comme un masque alpha ou un masque de luminance. Si aucune valeur n'est fournie et que la référence de masque est une image directe, elle serait considérée comme un masque alpha (ou) si la référence de masque est une URL, elle serait alors considérée comme un masque de luminance.
position	Cela spécifie la position de chaque couche de masque et son comportement est similaire à celui de la propriété `background-position` . La valeur peut être fournie dans une syntaxe à 1 valeur (comme `top` , `10%`) ou dans une syntaxe à 2 valeurs (comme en `top right` , `50% 50%`).
boîte à géométrie	Ceci spécifie la boîte sur laquelle le masque doit être écrêté (*zone de peinture du masque*) ou la boîte qui doit être utilisée comme référence pour l'origine du *masque* (*zone de positionnement du masque*) en fonction de la propriété. La liste des valeurs possibles est `content-box` , `padding-box` , `border-box` , `margin-box` , `fill-box` , `stroke-box` , `stroke-box view-box` . Des explications détaillées sur le fonctionnement de chacune de ces valeurs sont disponibles dans la spécification W3C .
taille bg	Cela représente la taille de chaque couche masque-image et a la même syntaxe que la `background-size` . La valeur peut être la longueur ou le pourcentage ou automatique ou couvrir ou contenir. La longueur, le pourcentage et l'auto peuvent être fournis en tant que valeur unique ou en tant que valeur unique pour chaque axe.
opérateur de compositing	Cela peut être n'importe lequel parmi `add` , `subtract` , `exclude` , `multiply` par couche et définit le type d'opération de composition à utiliser pour cette couche avec celles situées en dessous. Des explications détaillées sur chaque valeur sont disponibles dans les spécifications du W3C .

Remarques

Le découpage et le masquage CSS sont des concepts très récents et la prise en charge de ces propriétés par le navigateur est donc plutôt faible.

Masques:

Au moment de la rédaction (juillet 2016), Chrome, Safari et Opera supportaient ces propriétés avec le préfixe `-webkit-` .

Firefox ne nécessite pas de préfixes mais prend en charge les masques uniquement lorsqu'il est utilisé avec des éléments de `mask` SVG. Pour les éléments de `mask` SVG en ligne, la syntaxe est `mask: url(#msk)` alors que pour les éléments de `mask` dans un fichier SVG externe, la syntaxe est `mask: url('yourfilepath/yourfilename.svg#msk')` . `#msk` dans les deux cas fait référence à l' `id` de l'élément de `mask` auquel il est fait référence. Comme indiqué dans cette réponse , Firefox ne

prend actuellement en charge aucun paramètre autre que `mask-reference` dans la propriété `mask` .

Internet Explorer (et Edge) n'offre pas encore de support pour cette propriété.

La propriété `mask-mode` n'est actuellement pas prise en charge par les navigateurs **avec ou sans** préfixes.

Chemin de clip:

Au moment de rédiger (juillet 2016), Chrome, Safari et Opera supportent le `clip-path` lorsque le chemin est créé en utilisant des formes de base (comme `circle` , `polygon`) ou la syntaxe `url(#clipper)` avec SVG inline. Ils ne prennent pas en charge le découpage basé sur des formes faisant partie de fichiers SVG externes. De plus, le préfixe `-webkit` doit être présent.

Firefox ne prend en charge que la syntaxe `url()` pour le `clip-path` de `clip-path` tandis qu'Internet Explorer (et Edge) n'offre aucun support.

Examples

Clipping (Polygone)

CSS:

```
div{
  width:200px;
  height:200px;
  background:teal;
  clip-path: polygon(0 0, 0 100%, 100% 50%); /* refer remarks before usage */
}
```

HTML:

```
<div></div>
```

Dans l'exemple ci-dessus, un tracé de découpe **polygonal** est utilisé pour découper l'élément carré (200 x 200) en une forme de triangle. La forme de sortie est un triangle car le chemin commence à (c'est-à-dire que les premières coordonnées sont à) `0 0` - qui est le coin supérieur gauche de la boîte, puis passe à `0 100%` - qui est le coin inférieur gauche et puis finalement à `100% 50%` ce qui n'est rien d'autre que le centre droit de la boîte. Ces chemins sont auto-fermants (c'est-à-dire que le point de départ sera le point de fin) et que la forme finale est celle d'un triangle.

Cela peut également être utilisé sur un élément avec une image ou un dégradé en arrière-plan.

Voir l'exemple

Sortie:

Coupure (Cercle)

CSS:

```
div{
  width: 200px;
  height: 200px;
  background: teal;
  clip-path: circle(30% at 50% 50%); /* refer remarks before usage */
}
```

HTML

```
<div></div>
```

Cet exemple montre comment découper un div en cercle. L'élément est découpé dans un cercle dont le rayon est de 30% en fonction des dimensions de la boîte de référence, son centre étant situé au centre de la boîte de référence. Dans la mesure où aucun <clip-geometry-box> (en d'autres termes, zone de référence) n'est fourni, la zone de `border-box` de l'élément sera utilisée comme zone de référence.

La forme du cercle doit avoir un rayon et un centre avec des `(x,y)` :

```
circle(radius at x y)
```

Voir l'exemple

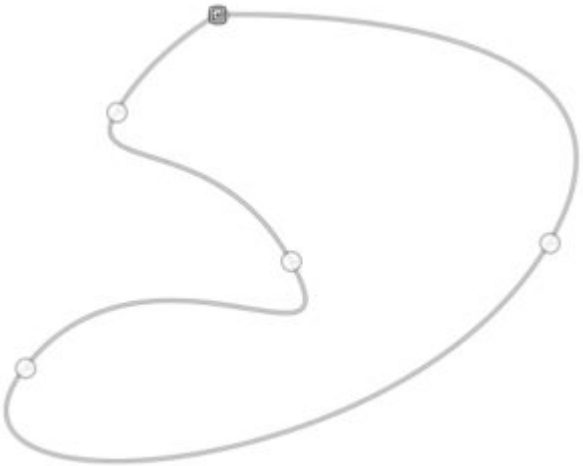
Sortie:

Clipping and Masking: Aperçu et différence

Avec **Clipping** and **Masking,** vous pouvez rendre certaines parties d'éléments transparentes ou opaques. Les deux peuvent être appliqués à n'importe quel élément HTML.

Coupure

Les clips sont des chemins vectoriels. En dehors de ce chemin, l'élément sera transparent, à l'intérieur il est opaque. Par conséquent, vous pouvez définir une propriété de `clip-path` sur les éléments. Chaque élément graphique existant également dans SVG peut être utilisé ici pour définir le chemin. Les exemples sont `circle()` , `polygon()` ou `ellipse()` .



Exemple

```
clip-path: circle(100px at center);
```

L'élément ne sera visible qu'à l'intérieur de ce cercle, situé au centre de l'élément et ayant un rayon de 100px.

Masquage

Les masques sont similaires aux clips, mais au lieu de définir un chemin, vous définissez un

masque sur les calques de l'élément. Vous pouvez imaginer ce masque comme une image composée principalement de deux couleurs: noir et blanc.

Luminance Masque : le noir signifie que la région est opaque et blanche qu'elle est transparente, mais il existe également une zone grise semi-transparente, ce qui vous permet d'effectuer des transitions régulières.

Alpha Mask : Seulement sur les zones transparentes du masque, l'élément sera opaque.

Cette image par exemple peut être utilisée comme masque de luminance pour faire pour un élément une transition très douce de droite à gauche et d'opaque à transparente.

La propriété `mask` vous permet de spécifier le type de masque et une image à utiliser comme calque.

Exemple

```
mask: url(masks.svg#rectangle) luminance;
```

Un élément appelé `rectangle` défini dans `masks.svg` sera utilisé comme **masque de luminance** sur l'élément.

Masque simple qui fait disparaître une image du solide au transparent

CSS

```
div {
  height: 200px;
  width: 200px;
  background: url(http://lorempixel.com/200/200/nature/1);
  mask-image: linear-gradient(to right, white, transparent);
}
```

HTML

```
<div></div>
```

Dans l'exemple ci-dessus, il y a un élément avec une image en arrière-plan. Le masque appliqué sur l'image (à l'aide de CSS) donne l'impression de disparaître de gauche à droite.

Le masquage est réalisé en utilisant un `linear-gradient` allant du blanc (à gauche) au transparent (à droite) en tant que masque. Comme il s'agit d'un masque alpha, l'image devient transparente là où le masque est transparent.

Sortie sans le masque:

Sortie avec le masque:

Remarque: Comme mentionné dans les remarques, l'exemple ci-dessus fonctionne uniquement avec Chrome, Safari et Opera lorsqu'il est utilisé avec le préfixe `-webkit` . Cet exemple (avec un `linear-gradient` tant qu'image de masque) n'est pas encore pris en charge dans Firefox.

Utiliser des masques pour couper un trou au milieu d'une image

CSS

```
div {
  width: 200px;
  height: 200px;
  background: url(http://lorempixel.com/200/200/abstract/6);
  mask-image: radial-gradient(circle farthest-side at center, transparent 49%, white 50%); /*
check remarks before using */
}
```

HTML

Dans l'exemple ci-dessus, un cercle transparent est créé au centre à l'aide d' `radial-gradient` et est ensuite utilisé comme masque pour produire l'effet d'un cercle découpé au centre d'une image.

Image sans masque:

Image avec masque:

Utilisation de masques pour créer des images de formes irrégulières

CSS

```
div { /* check remarks before usage */
  height: 200px;
  width: 400px;
  background-image: url(http://lorempixel.com/400/200/nature/4);
  mask-image: linear-gradient(to top right, transparent 49.5%, white 50.5%), linear-
gradient(to top left, transparent 49.5%, white 50.5%), linear-gradient(white, white);
  mask-size: 75% 25%, 25% 25%, 100% 75%;
  mask-position: bottom left, bottom right, top left;
  mask-repeat: no-repeat;
}
```

HTML

```
<div></div>
```

Dans l'exemple ci-dessus, trois images à `linear-gradient` (qui, lorsqu'elles sont placées dans leurs positions appropriées, couvrent 100% x 100% de la taille du conteneur) sont utilisées comme masques pour produire une coupe triangulaire transparente au bas de l'image.

Image sans le masque:

Image avec le masque:

Lire Couper et Masquer en ligne: https://riptutorial.com/fr/css/topic/3721/couper-et-masquer

Chapitre 16: Débordement

Syntaxe

- débordement: visible | caché | défiler | auto | initiale | hériter;

Paramètres

Valeur de `Overflow`	Détails
`visible`	Affiche tout le contenu débordant en dehors de l'élément
`scroll`	Masque le contenu débordant et ajoute une barre de défilement
`hidden`	Masque le contenu débordant, les deux barres de défilement disparaissent et la page devient fixe
`auto`	Identique au `scroll` si le contenu déborde, mais n'ajoute pas de barre de défilement si le contenu est adapté
`inherit`	Hériter la valeur de l'élément parent pour cette propriété

Remarques

> La propriété `overflow` spécifie s'il faut découper du contenu, afficher des barres de défilement ou étirer un conteneur pour afficher le contenu lorsqu'il déborde de son conteneur au niveau du bloc.

Lorsqu'un élément est trop petit pour afficher son contenu, que se passe-t-il? Par défaut, le contenu ne fera que déborder et s'afficher en dehors de l'élément. Cela rend votre travail mal en point. Vous voulez que votre travail soit beau, vous devez donc définir la propriété de débordement pour gérer le contenu débordant de manière souhaitable.

Les valeurs de la propriété `overflow` sont identiques à celles des propriétés `overflow-x` et `overflow-y` , sauf qu'elles s'appliquent le long de chaque axe

La propriété `overflow-wrap` a également été appelée propriété de `word-wrap` .

> Remarque importante: L' utilisation de la propriété de débordement avec une valeur différente de visible crée un nouveau contexte de mise en forme de bloc.

Examples

débordement: défilement

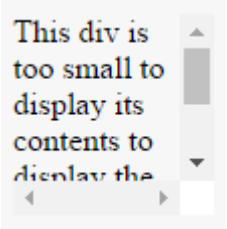
HTML

```
<div>
    This div is too small to display its contents to display the effects of the overflow
property.
</div>
```

CSS

```
div {
    width:100px;
    height:100px;
    overflow:scroll;
}
```

Résultat



Le contenu ci-dessus est découpé dans une boîte de 100 x 100 pixels, avec le défilement disponible pour afficher le contenu débordant.

La plupart des navigateurs de bureau affichent des barres de défilement horizontales et verticales, que le contenu soit tronqué ou non. Cela peut éviter des problèmes avec les barres de défilement apparaissant et disparaissant dans un environnement dynamique. Les imprimantes peuvent imprimer un contenu débordant.

débordement

`overflow-wrap` indique à un navigateur qu'il peut casser une ligne de texte à l'intérieur d'un élément ciblé sur plusieurs lignes dans un endroit incassable. Utile pour empêcher une longue chaîne de texte causant des problèmes de mise en page dus au débordement de son conteneur.

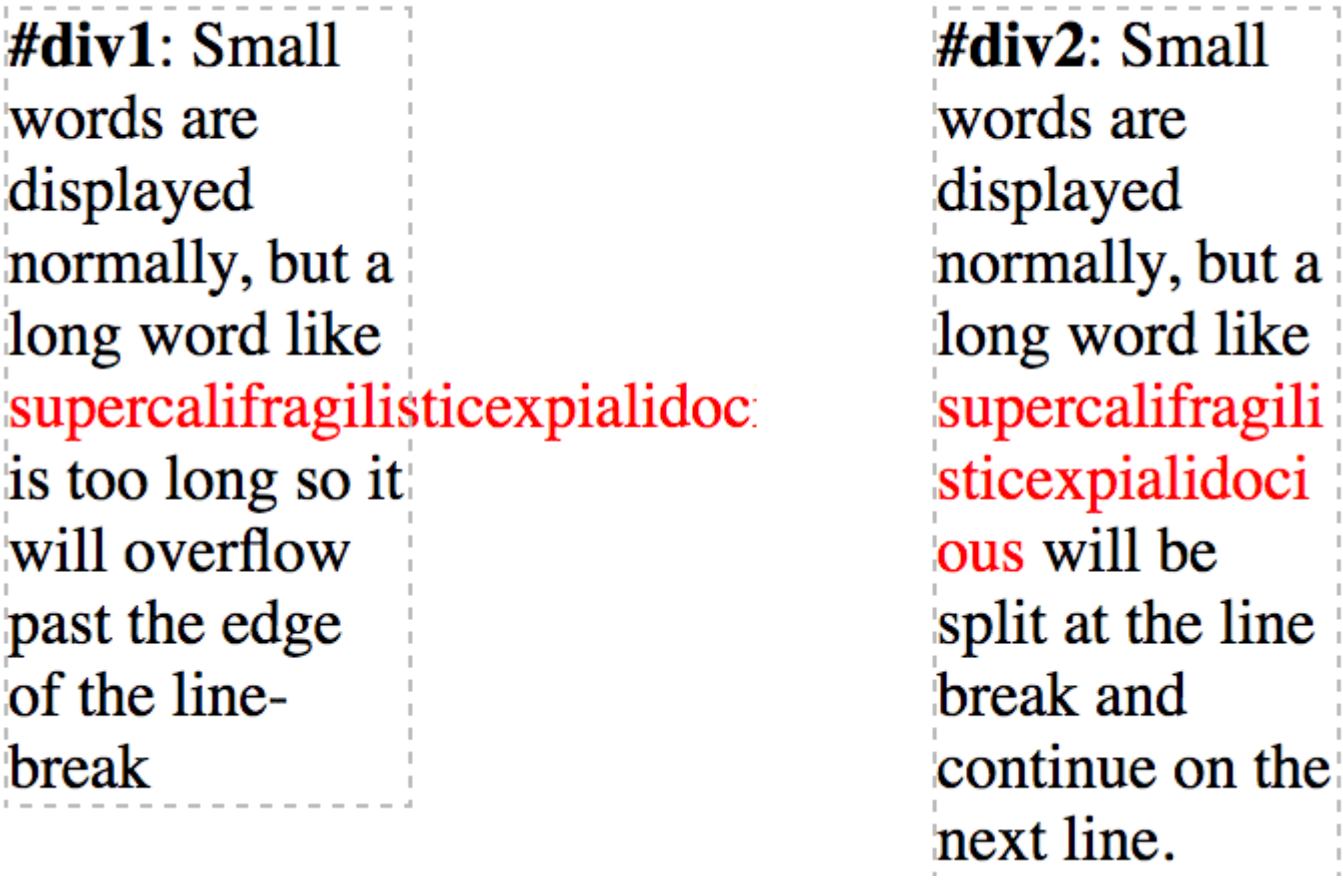
CSS

```
div {
    width:100px;
    outline: 1px dashed #bbb;
}

#div1 {
    overflow-wrap:normal;
}

#div2 {
```

```
    overflow-wrap:break-word;
}
```

HTML

```
<div id="div1">
      <strong>#div1</strong>: Small words are displayed normally, but a long word like <span
style="red;">supercalifragilisticexpialidocious</span> is too long so it will overflow past
the edge of the line-break
</div>

<div id="div2">
      <strong>#div2</strong>: Small words are displayed normally, but a long word like <span
style="red;">supercalifragilisticexpialidocious</span> will be split at the line break and
continue on the next line.
</div>
```



`overflow-wrap` - Valeur	Détails
`normal`	Permet un dépassement de mot s'il est plus long que la ligne
`break-word`	Séparera un mot en plusieurs lignes, si nécessaire
`inherit`	Hérite la valeur de l'élément parent pour cette propriété

débordement: visible

HTML

```
<div>
     Even if this div is too small to display its contents, the content is not clipped.
</div>
```

CSS

```
div {
     width:50px;
     height:50px;
     overflow:visible;
}
```

Résultat

Le contenu n'est pas tronqué et sera rendu hors de la zone de contenu s'il dépasse sa taille de conteneur.

Contexte de mise en forme de bloc créé avec un dépassement de capacité

L'utilisation de la propriété `overflow` avec une valeur différente de `visible` créera un nouveau **contexte de mise en forme de bloc** . Ceci est utile pour aligner un élément de bloc à côté d'un élément flottant.

CSS

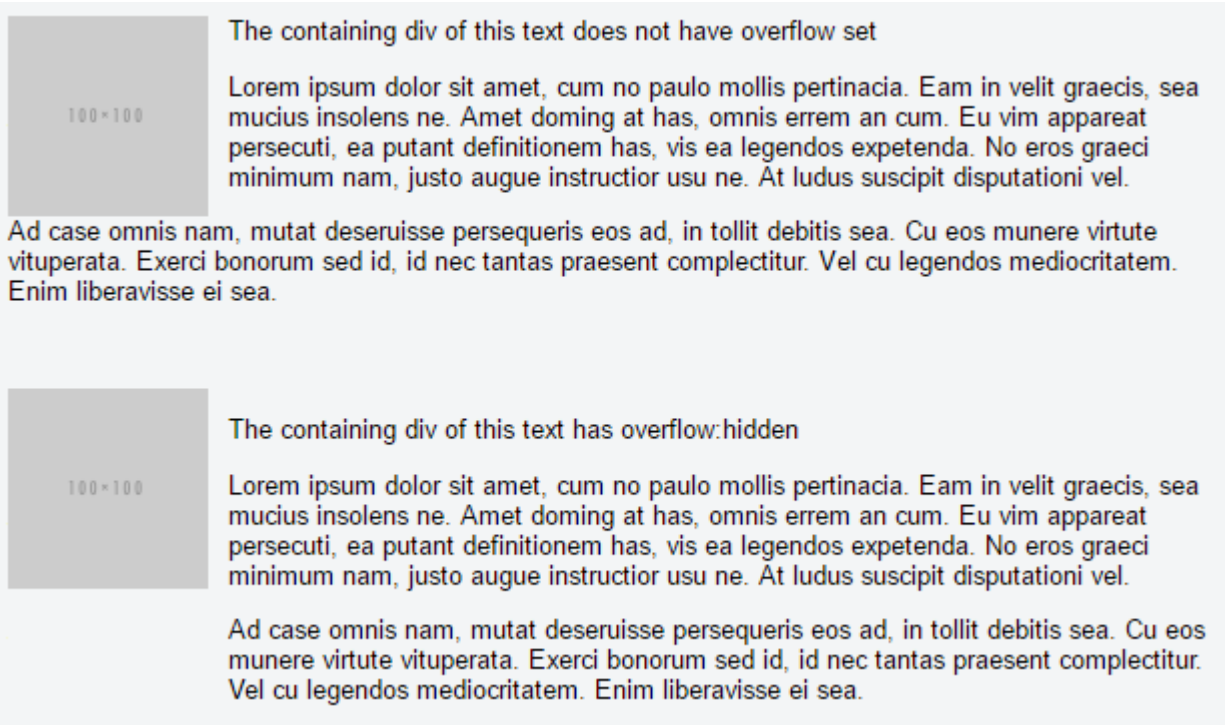
```
img {
    float:left;
    margin-right: 10px;
}
 div {
     overflow:hidden; /* creates block formatting context */
}
```

HTML

```
<img src="http://placehold.it/100x100">
```

```
<div>
  <p>Lorem ipsum dolor sit amet, cum no paulo mollis pertinacia.</p>
  <p>Ad case omnis nam, mutat deseruisse persequeris eos ad, in tollit debitis sea.</p>
</div>
```

Résultat



Cet exemple montre comment les paragraphes d'un div avec le jeu de propriétés de `overflow` interagiront avec une image flottante.

débordement-x et débordement-y

Ces deux propriétés fonctionnent de la même manière que la propriété `overflow` et acceptent les mêmes valeurs. Le paramètre `overflow-x` ne fonctionne que sur l'axe x ou de gauche à droite. Le `overflow-y` fonctionne sur l'axe y ou de haut en bas.

HTML

```
<div id="div-x">
    If this div is too small to display its contents,
    the content to the left and right will be clipped.
</div>

<div id="div-y">
    If this div is too small to display its contents,
    the content to the top and bottom will be clipped.
</div>
```

CSS

```
div {
    width: 200px;
```

```
    height: 200px;
}

#div-x {
    overflow-x: hidden;
}

#div-y {
    overflow-y: hidden;
}
```

Lire Débordement en ligne: https://riptutorial.com/fr/css/topic/4947/debordement

Chapitre 17: Des animations

Syntaxe

- `transition: <property> <duration> <timing-function> <delay>;`
- `@keyframes <identifier>`
- `[ [ from | to | <percentage> ] [, from | to | <percentage> ]* block ]*`

Paramètres

Transition	
Paramètre	Détails
propriété	Soit la propriété CSS sur laquelle passer, ou `all` , qui spécifie toutes les propriétés pouvant être transposées.
durée	Temps de transition, en secondes ou en millisecondes.
fonction de synchronisation	Spécifie une fonction pour définir comment les valeurs intermédiaires des propriétés sont calculées. Les valeurs communes sont la `ease` , la `linear` et la `step-end` . Découvrez la feuille d' accélération de la fonction d'accélération pour en savoir plus.
retard	Temps d'attente en secondes ou en millisecondes avant la lecture de l'animation.
@ keyframes	
[de | à | `<percentage>`]	Vous pouvez soit spécifier une heure définie avec une valeur en pourcentage, soit deux valeurs de pourcentage, à savoir `10%, 20%` , pour une période de temps pendant laquelle les attributs définis par l'image clé sont définis.
`block`	Toute quantité d'attributs CSS pour l'image clé.

Examples

Animations avec la propriété de transition

Utile pour les animations simples, la propriété de `transition` CSS permet aux propriétés CSS basées sur les nombres de s'animer entre les états.

Exemple

```
.Example{
    height: 100px;
    background: #fff;
}

.Example:hover{
    height: 120px;
    background: #ff0000;
}
```

Voir résultat

Par défaut, si vous `.Example` un élément avec la classe `.Example` la hauteur de l'élément passe immédiatement à `120px` et sa couleur d'arrière-plan `120px` rouge (`#ff0000`).

En ajoutant la propriété de `transition` , nous pouvons provoquer ces modifications au fil du temps:

```
.Example{
    ...
    transition: all 400ms ease;
}
```

Voir résultat

La valeur `all` applique la transition à toutes les propriétés compatibles (basées sur les nombres). Tout nom de propriété compatible (tel que `height` ou `top`) peut être substitué à ce mot-clé.

`400ms` spécifie la durée de la transition. Dans ce cas, le changement de hauteur de l'élément prendra 400 millisecondes.

Enfin, la valeur `ease` est la fonction d'animation, qui détermine la manière dont l'animation est jouée. `ease` signifie commencer lentement, accélérer, puis terminer à nouveau lentement. Les autres valeurs sont `linear` , `ease-out` et `ease-in` .

Compatibilité entre navigateurs

La propriété de `transition` est généralement bien prise en charge sur tous les principaux navigateurs, à l'exception d'IE 9. Pour les versions antérieures des navigateurs basés sur Firefox et Webkit, utilisez les préfixes du fournisseur comme suit:

```
.Example{
    transition:         all 400ms ease;
    -moz-transition:    all 400ms ease;
    -webkit-transition: all 400ms ease;
}
```

Remarque: La propriété de `transition` peut animer les modifications entre deux valeurs numériques, indépendamment de l'unité. Il peut également faire la transition entre les unités, telles que `100px` à `50vh` . Cependant, il ne peut pas faire la transition entre un nombre et une valeur par défaut ou automatique, telle que la transition d'une hauteur d'élément de `100px` à `auto` .

Augmentation des performances d'animation à l'aide de l'attribut `will-change`

Lors de la création d'animations et d'autres actions lourdes de GPU, il est important de comprendre l'attribut `will-change` .

Les images clés CSS et la propriété de `transition` utilisent l'accélération GPU. Les performances sont augmentées par le déchargement des calculs sur le GPU du périphérique. Cela se fait en créant des couches de peinture (parties de la page qui sont rendues individuellement) qui sont déchargées sur le GPU pour être calculées. La propriété `will-change` indique au navigateur ce qui va animer, permettant au navigateur de créer des zones de peinture plus petites, augmentant ainsi les performances.

La propriété `will-change` accepte une liste de propriétés séparées par des virgules à animer. Par exemple, si vous envisagez de transformer un objet et de modifier son opacité, vous devez spécifier:

```
.Example{
    ...
    will-change: transform, opacity;
}
```

Remarque: Utilisez `will-change` avec modération. La définition de `will-change` pour chaque élément d'une page peut entraîner des problèmes de performances, car le navigateur peut tenter de créer des couches de peinture pour chaque élément, ce qui augmente considérablement la quantité de traitement effectuée par le GPU.

Animations avec images clés

Pour les animations CSS multi-étapes, vous pouvez créer CSS `@keyframes` . Les images clés vous permettent de définir plusieurs points d'animation, appelés images clés, pour définir des animations plus complexes.

Exemple de base

Dans cet exemple, nous allons créer une animation d'arrière-plan de base qui passe de toutes les couleurs.

```
@keyframes rainbow-background {
          0%     { background-color: #ff0000; }
      8.333%     { background-color: #ff8000; }
     16.667%     { background-color: #ffff00; }
     25.000%     { background-color: #80ff00; }
     33.333%     { background-color: #00ff00; }
```

```
    41.667%     { background-color: #00ff80; }
    50.000%     { background-color: #00ffff; }
    58.333%     { background-color: #0080ff; }
    66.667%     { background-color: #0000ff; }
    75.000%     { background-color: #8000ff; }
    83.333%     { background-color: #ff00ff; }
    91.667%     { background-color: #ff0080; }
    100.00%     { background-color: #ff0000; }
}

.RainbowBackground {
    animation: rainbow-background 5s infinite;
}
```

Voir résultat

Il y a plusieurs choses à noter ici. Tout d'abord, la syntaxe `@keyframes` réelle.

```
@keyframes rainbow-background{
```

Cela définit le nom de l'animation sur l' `rainbow-background` .

```
0%     { background-color: #ff0000; }
```

C'est la définition d'une image clé dans l'animation. La première partie, le `0%` dans le cas, définit où se trouve l'image-clé pendant l'animation. Le `0%` implique que c'est 0% du temps d'animation total depuis le début.

L'animation passera automatiquement entre les images clés. Ainsi, en définissant la couleur d'arrière-plan suivante à `8.333%` , l'animation prendra 8,333% du temps de transition entre ces images clés.

```
.RainbowBackground {
    animation: rainbow-background 5s infinite;
}
```

Ce code joint notre animation à tous les éléments qui ont la classe `.RainbowBackground` .

La propriété d'animation réelle prend les arguments suivants.

- **nom-animation** : le nom de notre animation. Dans ce cas, `rainbow-background`
- **animation-duration** : **durée de** l'animation, dans ce cas 5 secondes.
- **animation-iteration-count (Facultatif)** : nombre de fois que l'animation sera mise en boucle. Dans ce cas, l'animation se poursuivra indéfiniment. Par défaut, l'animation sera jouée une fois.
- **animation-delay (Facultatif)** : Spécifie combien de temps attendre avant le début de l'animation. La valeur par défaut est 0 seconde et peut prendre des valeurs négatives. Par exemple, `-2s` lancerait l'animation 2 secondes dans sa boucle.
- **animation-timing-function (Facultatif)** : Spécifie la courbe de vitesse de l'animation. Par défaut, cela `ease` , où l'animation commence lentement, devient plus rapide et se termine lentement.

Dans cet exemple particulier, les images clés `0%` et `100%` spécifient `{ background-color: #ff0000; }`. Partout où deux images clés ou plus partagent un état, on peut les spécifier dans une seule déclaration. Dans ce cas, les deux lignes `0%` et `100%` pourraient être remplacées par cette seule ligne:

```
0%, 100%     { background-color: #ff0000; }
```

Compatibilité entre navigateurs

Pour les navigateurs WebKit plus anciens, vous devez utiliser le préfixe du fournisseur dans la déclaration `@keyframes` et la propriété `animation`, comme ceci:

```
@-webkit-keyframes{}

-webkit-animation: ...
```

Exemples de syntaxe

Notre premier exemple de syntaxe montre la propriété abrégée d'animation en utilisant toutes les propriétés / paramètres disponibles:

```
  animation: 3s         ease-in           1s      2                reverse     both
paused       slidein;
  /*         duration | timing-function | delay | iteration-count | direction | fill-mode |
play-state | name   */
```

Notre deuxième exemple est un peu plus simple et montre que certaines propriétés peuvent être omises:

```
  animation: 3s         linear            1s      slidein;
  /*         duration | timing-function | delay | name   */
```

Notre troisième exemple montre la déclaration la plus minimale. Notez que le nom de l'animation et la durée de l'animation doivent être déclarés:

```
  animation: 3s         slidein;
  /*         duration | name */
```

Il convient également de mentionner que, lorsque vous utilisez l'animation abrégée, l'ordre des propriétés fait la différence. De toute évidence, le navigateur peut confondre votre durée avec votre retard.

Si la concision n'est pas votre truc, vous pouvez également ignorer la propriété abrégée et écrire chaque propriété individuellement:

```
animation-duration: 3s;
```

```
animation-timing-function: ease-in;
animation-delay: 1s;
animation-iteration-count: 2;
animation-direction: reverse;
animation-fill-mode: both;
animation-play-state: paused;
animation-name: slidein;
```

Lire Des animations en ligne: https://riptutorial.com/fr/css/topic/590/des-animations

Chapitre 18: Des flotteurs

Syntaxe

- clear: none | à gauche | droit | les deux inline-start | fin en ligne;
- float: gauche | droit | aucun inline-start | fin en ligne;

Remarques

> Comme float implique l'utilisation de la disposition des blocs, il modifie la valeur calculée des valeurs d'affichage dans certains cas [1]

[1]: https://developer.mozilla.org/en-US/docs/Web/CSS/float MDN

Examples

Flotter une image dans un texte

L'utilisation la plus élémentaire d'un flotteur consiste à placer un texte autour d'une image. Le code ci-dessous produira deux paragraphes et une image, le second paragraphe circulant autour de l'image. Notez qu'il est toujours contenu *après* l'élément flottant qui circule autour de l'élément flottant.

HTML:

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet. Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa. Vestibulum lacinia arcu eget nulla. </p>

<img src="http://lorempixel.com/200/100/" />

<p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. </p>
```

CSS:

```
img {
  float:left;
  margin-right:1rem;
}
```

Ce sera la sortie

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet. Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa. Vestibulum lacinia arcu eget nulla.

Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet.

Codepen Link

Mise en page simple de deux colonnes à largeur fixe

Une disposition simple à deux colonnes se compose de deux éléments flottants à largeur fixe. Notez que la barre latérale et la zone de contenu ne sont pas de la même hauteur dans cet exemple. Il s'agit de l'une des parties délicates des mises en page à plusieurs colonnes utilisant des flottants, et nécessite des solutions pour que plusieurs colonnes aient la même hauteur.

HTML:

```
<div class="wrapper">

<div class="sidebar">
  <h2>Sidebar</h2>

  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio.</p>
</div>

<div class="content">
  <h1>Content</h1>

  <p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos
himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor.
Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis
tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel,
suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. </p>
</div>

</div>
```

CSS:

```
.wrapper {
  width:600px;
  padding:20px;
  background-color:pink;

  /* Floated elements don't use any height. Adding "overflow:hidden;" forces the
     parent element to expand to contain its floated children. */
  overflow:hidden;
}

.sidebar {
  width:150px;
  float:left;
  background-color:blue;
}

.content {
  width:450px;
  float:right;
  background-color:yellow;
}
```

Trois colonnes simples à largeur fixe

HTML:

```
<div class="wrapper">
  <div class="left-sidebar">
    <h1>Left Sidebar</h1>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. </p>
  </div>
  <div class="content">
    <h1>Content</h1>
    <p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos
himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor.
Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis
```

```
tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel,
suscipit quis, luctus non, massa. </p>
  </div>
  <div class="right-sidebar">
    <h1>Right Sidebar</h1>
    <p>Fusce ac turpis quis ligula lacinia aliquet.</p>
  </div>
</div>
```

CSS:

```
.wrapper {
  width:600px;
  background-color:pink;
  padding:20px;

  /* Floated elements don't use any height. Adding "overflow:hidden;" forces the
     parent element to expand to contain its floated children. */
  overflow:hidden;
}

.left-sidebar {
  width:150px;
  background-color:blue;
  float:left;
}

.content {
  width:300px;
  background-color:yellow;
  float:left;
}

.right-sidebar {
  width:150px;
  background-color:green;
  float:right;
}
```

Mise en page paresseuse / gourmande à deux colonnes

Cette disposition utilise une colonne flottante pour créer une disposition à deux colonnes sans largeur définie. Dans cet exemple, la barre latérale gauche est "paresseuse", en ce sens qu'elle ne prend que l'espace nécessaire. Une autre façon de dire ceci est que la barre latérale gauche est "emballée sous film rétractable". La bonne colonne de contenu est "gourmande", car elle occupe tout l’espace restant.

HTML:

```
<div class="sidebar">
<h1>Sidebar</h1>
<img src="http://lorempixel.com/150/200/" />
</div>

<div class="content">
<h1>Content</h1>
```

```
<p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet. Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa. Vestibulum lacinia arcu eget nulla. </p>
<p>Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. Sed dignissim lacinia nunc. Curabitur tortor. Pellentesque nibh. Aenean quam. In scelerisque sem at dolor. Maecenas mattis. Sed convallis tristique sem. Proin ut ligula vel nunc egestas porttitor. Morbi lectus risus, iaculis vel, suscipit quis, luctus non, massa. Fusce ac turpis quis ligula lacinia aliquet. Mauris ipsum. Nulla metus metus, ullamcorper vel, tincidunt sed, euismod in, nibh. </p>
</div>
```

CSS:

```
.sidebar {
  /* `display:table;` shrink-wraps the column */
  display:table;
  float:left;
  background-color:blue;
}

.content {
  /* `overflow:hidden;` prevents `.content` from flowing under `.sidebar` */
  overflow:hidden;
  background-color:yellow;
}
```

Violon

propriété claire

La propriété clear est directement liée aux flottants. Valeurs de propriété:

- none - Par défaut Permet des éléments flottants des deux côtés
- à gauche - Aucun élément flottant autorisé à gauche
- right - Aucun élément flottant autorisé à droite
- les deux - Aucun élément flottant n'est autorisé du côté gauche ou du côté droit
- initial - Définit cette propriété sur sa valeur par défaut. Lisez à propos de l'initiale
- inherit - Hérite cette propriété de son élément parent. En savoir plus sur hériter

```
<html>
<head>
<style>
img {
    float: left;
}

p.clear {
    clear: both;
}
</style>
</head>
<body>

<img src="https://static.pexels.com/photos/69372/pexels-photo-69372-medium.jpeg" width="100">
```

```
<p>Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum
Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum </p>
<p class="clear">Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum
Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum Lorem ipsoum </p>

</body>
</html>
```

Clearfix

> Le hack clearfix est un moyen populaire de contenir des flottants (N. Gallagher aka @necolas)

Ne pas confondre avec la propriété `clear` , clearfix est un *concept* (qui est également lié aux flottants, donc à la confusion possible). Pour *contenir des flottants* , vous devez ajouter la classe `.cf` ou `.clearfix` au conteneur (**le parent**) et mettre en forme cette classe avec quelques règles décrites ci-dessous.

3 versions avec des effets légèrement différents (sources: un nouveau hack micro clearfix de N. Gallagher et clearfix rechargé par TJ Koblentz):

Clearfix (avec la marge supérieure effondrant les flotteurs contenus)

```
.cf:after {
    content: "";
    display: table;
}

.cf:after {
    clear: both;
}
```

Clearfix empêche également l'effondrement de la marge supérieure des flotteurs contenus

```
/**
 * For modern browsers
 * 1. The space content is one way to avoid an Opera bug when the
 *    contenteditable attribute is included anywhere else in the document.
 *    Otherwise it causes space to appear at the top and bottom of elements
 *    that are clearfixed.
 * 2. The use of `table` rather than `block` is only necessary if using
 *    `:before` to contain the top-margins of child elements.
 */
```

```
.cf:before,
.cf:after {
    content: " "; /* 1 */
    display: table; /* 2 */
}

.cf:after {
    clear: both;
}
```

Clearfix avec le support des navigateurs périmés IE6 et IE7

```
.cf:before,
.cf:after {
    content: " ";
    display: table;
}

.cf:after {
    clear: both;
}

/**
 * For IE 6/7 only
 * Include this rule to trigger hasLayout and contain floats.
 */
.cf {
    *zoom: 1;
}
```

Codepen montrant un effet de clearfix

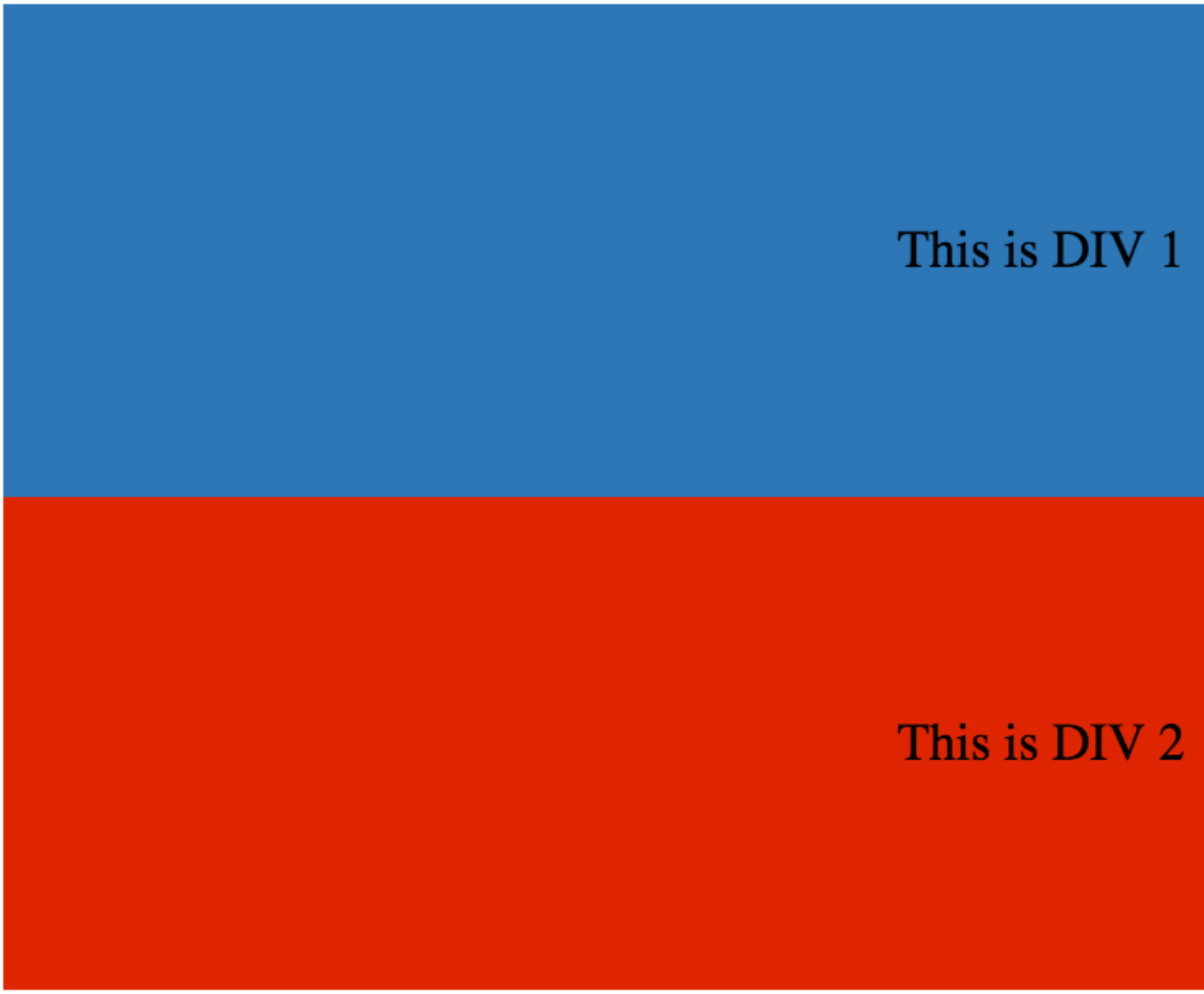
Autre ressource: tout ce que vous savez sur clearfix est incorrect (clearfix et BFC - Context de formatage de bloc alors que hasLayout concerne les navigateurs périmés IE6 7)

DIV en ligne utilisant un flottant

Le `div` est un élément de niveau bloc, c'est-à-dire qu'il occupe toute la largeur de la page et que les frères et soeurs sont placés les uns en dessous des autres, quelle que soit leur largeur.

```
<div>
    <p>This is DIV 1</p>
</div>
<div>
    <p>This is DIV 2</p>
</div>
```

La sortie du code suivant sera



Nous pouvons les rendre en ligne en ajoutant une propriété css `float` à la `div` .

HTML:

```
<div class="outer-div">
    <div class="inner-div1">
        <p>This is DIV 1</p>
    </div>
    <div class="inner-div2">
        <p>This is DIV 2</p>
    </div>
</div>
```

CSS

```
.inner-div1 {
    width: 50%;
    margin-right:0px;
    float:left;
```

```
    background : #337ab7;
    padding:50px 0px;
}

.inner-div2 {
    width: 50%;
    margin-right:0px;
    float:left;
    background : #dd2c00;
    padding:50px 0px;
}

p {
    text-align:center;
}
```

Codepen Link

Utilisation de la propriété de débordement pour effacer les flottants

Définir la valeur de `overflow` sur `hidden` , `auto` ou `scroll` jusqu'à un élément effacera tous les flottants de cet élément.

Note: en utilisant `overflow:scroll` affichera toujours la scrollbox

Lire Des flotteurs en ligne: https://riptutorial.com/fr/css/topic/405/des-flotteurs

Chapitre 19: Disposition de la boîte flexible (Flexbox)

Introduction

Le module Flexible Box, ou simplement «flexbox» en abrégé, est un modèle de boîte conçu pour les interfaces utilisateur. Il permet aux utilisateurs d'aligner et de répartir l'espace entre les éléments d'un conteneur de façon tailles d'écran. Un conteneur flexible étend les éléments pour remplir l'espace disponible et les réduit pour éviter tout débordement.

Syntaxe

- affichage: flex;
- flex-direction: row | ligne inverse | colonne | colonne inversée;
- flex-wrap: nowrap | envelopper | wrap-reverse;
- flex-flow: <'flex-direction'> || <'flex-wrap'>
- Justification-content: flex-start | flex-end | centre | espace-entre | espace autour
- align-items: flex-start | flex-end | centre | ligne de base | étendue;
- align-content: flex-start | flex-end | centre | espace-entre | espace autour | étendue;
- ordre: <entier>;
- flex-grow: <nombre>; / * par défaut 0 * /
- flex-shrink: <numéro>; / * par défaut 1 * /
- base flexible: <longueur> | auto; / * auto par défaut * /
- flex: none | [<'flex-grow'> <'flex-shrink'> "? || <'flex-basis'>]
- align-self: auto | flex-start | flex-end | centre | ligne de base | étendue;

Remarques

Préfixes Vender

- display: -webkit-box; / * Chrome <20 * /
- affichage: -webkit-flex; / * Chrome 20+ * /
- display: -moz-box; / * Firefox * /
- afficher: -ms-flexbox; /* C'EST À DIRE */
- affichage: flex; / * Navigateurs modernes * /

Ressources

- Un guide complet pour Flexbox
- Résolu par Flexbox
- Qu'est-ce que la Flexbox?!

- Flexbox en 5 minutes
- Flexbugs

Examples

Pied collant à hauteur variable

Ce code crée un bas de page collant. Lorsque le contenu n'atteint pas la fin de la fenêtre, le pied de page reste en bas de la fenêtre d'affichage. Lorsque le contenu dépasse le bas de la fenêtre, le pied de page est également sorti de la fenêtre d'affichage. *Voir résultat*

HTML:

```
<div class="header">
  <h2>Header</h2>
</div>

<div class="content">
  <h1>Content</h1>
  <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Integer nec odio. Praesent
libero. Sed cursus ante dapibus diam. Sed nisi. Nulla quis sem at nibh elementum imperdiet.
Duis sagittis ipsum. Praesent mauris. Fusce nec tellus sed augue semper porta. Mauris massa.
Vestibulum lacinia arcu eget nulla. Class aptent taciti sociosqu ad litora torquent per
conubia nostra, per inceptos himenaeos. Curabitur sodales ligula in libero. </p>
</div>

<div class="footer">
  <h4>Footer</h4>
</div>
```

CSS:

```
html, body {
  height: 100%;
}

body {
  display: flex;
  flex-direction: column;
}

.content {
  /* Include `0 auto` for best browser compatibility. */
  flex: 1 0 auto;
}

.header, .footer {
  background-color: grey;
  color: white;
  flex: none;
}
```

Holy Grail Layout utilisant Flexbox

La disposition de Holy Graal est une disposition avec un en-tête et un pied de page de hauteur fixe, et un centre avec 3 colonnes. Les 3 colonnes incluent un sidenav de largeur fixe, un centre fluide et une colonne pour d'autres contenus comme les annonces (le centre fluide apparaît en premier dans le balisage). CSS Flexbox peut être utilisé pour y parvenir avec un balisage très simple:

Balisage HTML:

```
<div class="container">
  <header class="header">Header</header>
  <div class="content-body">
    <main class="content">Content</main>
    <nav class="sidenav">Nav</nav>
    <aside class="ads">Ads</aside>
  </div>
  <footer class="footer">Footer</footer>
</div>
```

CSS:

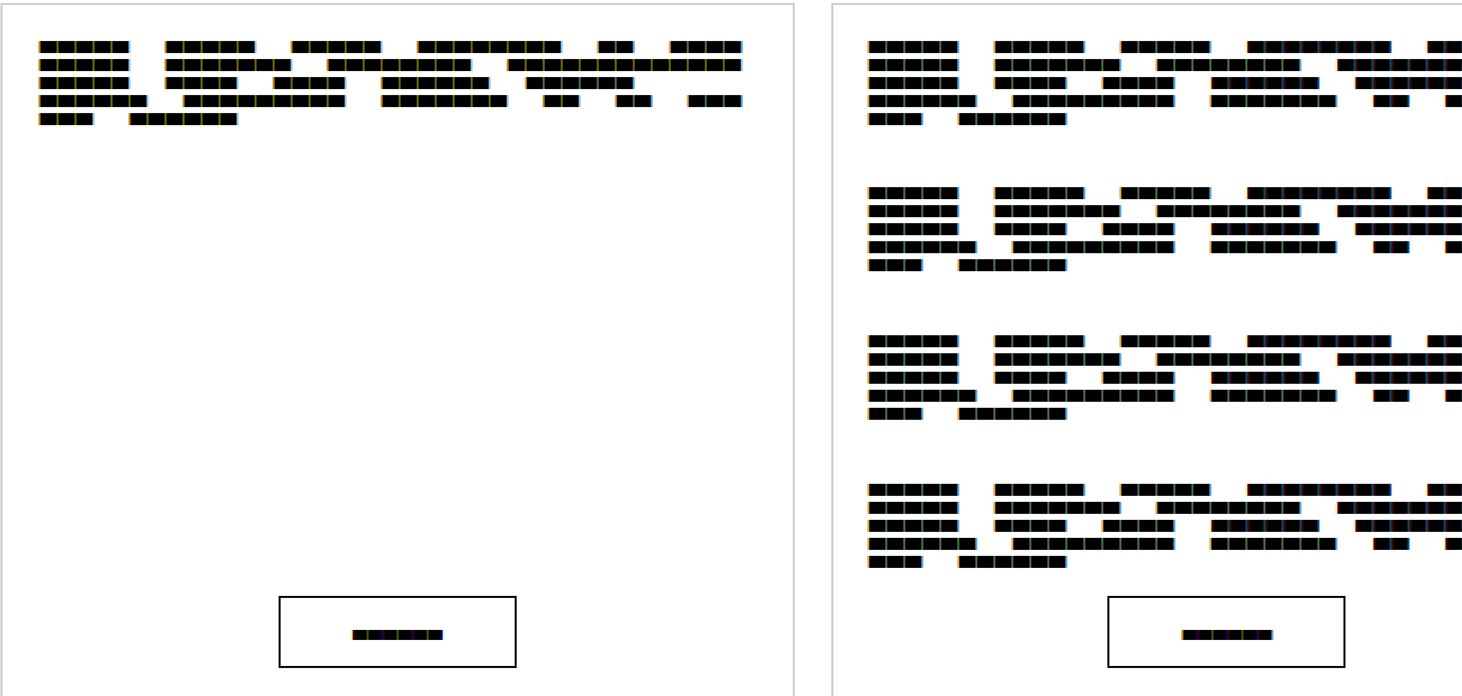
```
body {
  margin: 0;
  padding: 0;
}

.container {
  display: flex;
  flex-direction: column;
  height: 100vh;
}

.header {
  flex: 0 0 50px;
}

.content-body {
  flex: 1 1 auto;

  display: flex;
  flex-direction: row;
}

.content-body .content {
  flex: 1 1 auto;
  overflow: auto;
}

.content-body .sidenav {
  order: -1;
  flex: 0 0 100px;
  overflow: auto;
}

.content-body .ads {
  flex: 0 0 100px;
  overflow: auto;
}
```

```
.footer {
  flex: 0 0 50px;
}
```

Démo

Boutons parfaitement alignés à l'intérieur des cartes avec flexbox

De nos jours, il s'agit d'un modèle de conception consistant à aligner verticalement l' **appel à des actions à l'** intérieur de ses cartes contenant les éléments suivants:



Cela peut être réalisé en utilisant un tour spécial avec `flexbox`

HTML

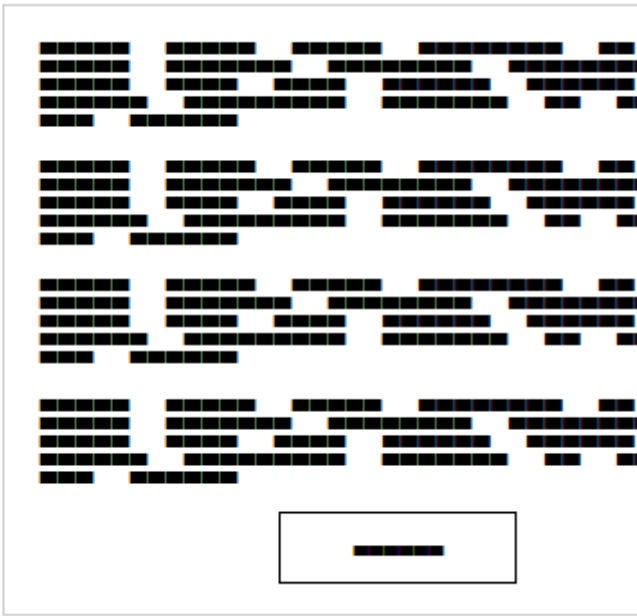
```
<div class="cards">
  <div class="card">
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p><button>Action</button></p>
  </div>
  <div class="card">
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p>Lorem ipsum Magna proident ex anim dolor ullamco pariatur reprehenderit culpa esse enim
mollit labore dolore voluptate ullamco et ut sed qui minim.</p>
    <p><button>Action</button></p>
  </div>
</div>
```

Tout d'abord, nous utilisons CSS pour appliquer l' `display: flex;` au conteneur. Cela va créer 2 colonnes égales en hauteur avec le contenu qui circule naturellement à l'intérieur

CSS

```
.cards {
    display: flex;
}
.card {
    border: 1px solid #ccc;
    margin: 10px 10px;
    padding: 0 20px;
}
button {
    height: 40px;
    background: #fff;
    padding: 0 40px;
    border: 1px solid #000;
}
p:last-child {
    text-align: center;
}
```

La mise en page changera et deviendra comme ceci:



Pour déplacer les boutons au bas du bloc, nous devons appliquer l' `display: flex;` à la carte elle-même avec la direction définie sur `column` . Après cela, nous devrions sélectionner le dernier élément à l'intérieur de la carte et définir le `margin-top` sur `auto` . Cela poussera le dernier paragraphe au bas de la carte et atteindra le résultat requis.

CSS final:

```
.cards {
    display: flex;
}
```

```
.card {
    border: 1px solid #ccc;
    margin: 10px 10px;
    padding: 0 20px;
    display: flex;
    flex-direction: column;
}
button {
    height: 40px;
    background: #fff;
    padding: 0 40px;
    border: 1px solid #000;
}
p:last-child {
    text-align: center;
    margin-top: auto;
}
```

Centrage vertical et horizontal dynamique (éléments d'alignement, contenu justifié)

Exemple simple (centrage d'un seul élément)

HTML

```
<div class="aligner">
  <div class="aligner-item">…</div>
</div>
```

CSS

```
.aligner {
  display: flex;
  align-items: center;
  justify-content: center;
}

.aligner-item {
  max-width: 50%; /*for demo. Use actual width instead.*/
}
```

Voici une démo .

Raisonnement

Propriété	Valeur	La description
`align-`	`center`	Cela centre les éléments le long de l'axe autre que celui spécifié par la

Propriété	Valeur	La description
`items`		`flex-direction` de `flex-direction` , c'est `flex-direction` dire le centrage vertical pour une boîte de flexion horizontale et le centrage horizontal pour une boîte de flexion verticale.
`justify-content`	`center`	Cela centre les éléments le long de l'axe spécifié par `flex-direction` . C'est-à-dire que, pour un flexbox horizontal (`flex-direction: row`), il se centre horizontalement, et pour un flexbox vertical (`flex-direction: column`) flexbox, il se centre verticalement)

Exemples de propriétés individuelles

Tous les styles ci-dessous sont appliqués sur cette mise en page simple:

```
<div id="container">
  <div></div>
  <div></div>
  <div></div>
</div>
```

où `#container` est le `flex-box` .

Exemple: `justify-content: center` sur un flexbox horizontal

CSS:

```
div#container {
  display: flex;
  flex-direction: row;
  justify-content: center;
}
```

Résultat:

Voici une démo .

Exemple: `justify-content: center` sur un flexbox vertical

CSS:

```
div#container {
  display: flex;
  flex-direction: column;
  justify-content: center;
}
```

Résultat:

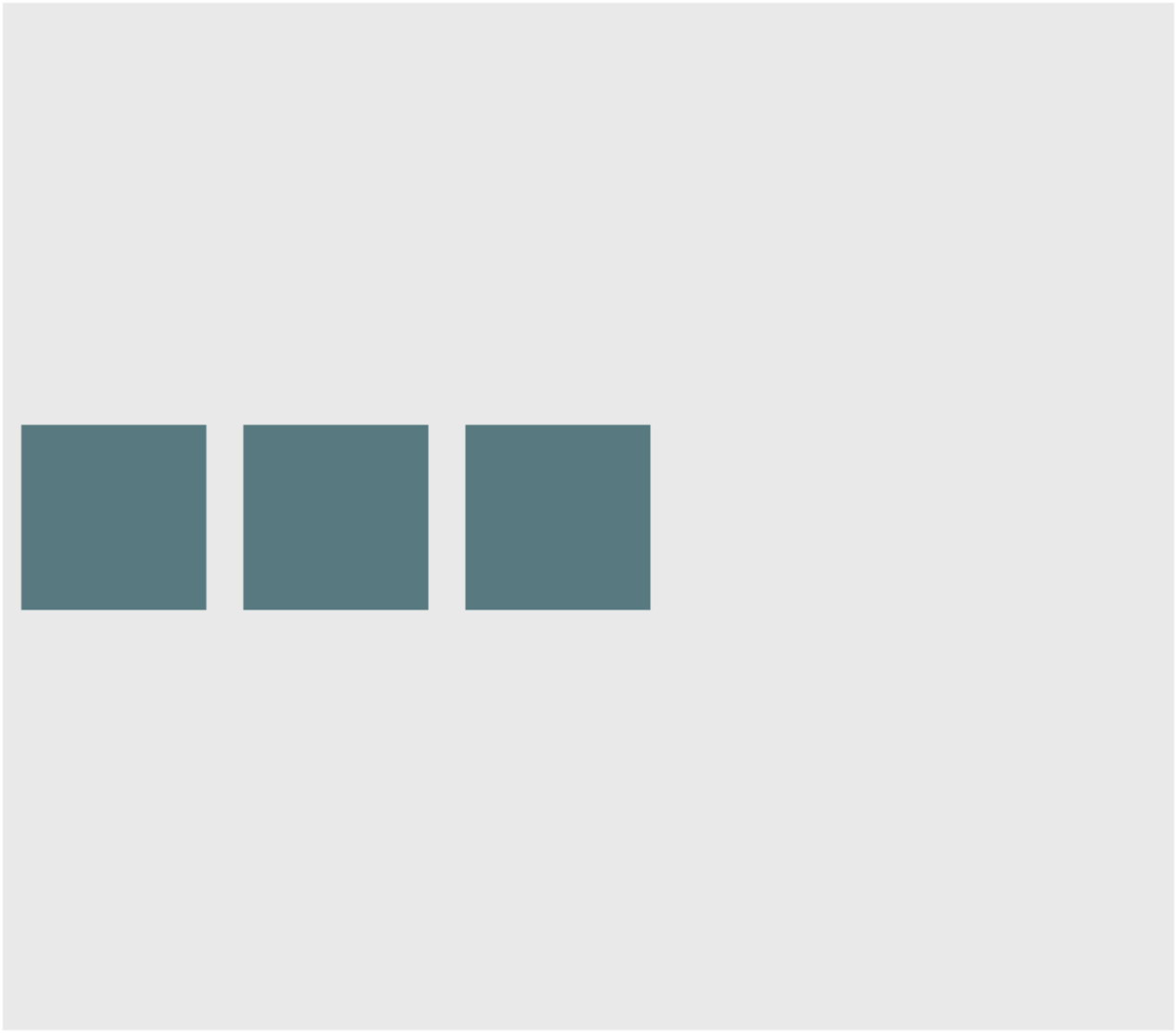
Voici une démo .

Exemple: `align-content: center` sur un flexbox horizontal

CSS:

```
div#container {
  display: flex;
  flex-direction: row;
  align-items: center;
}
```

Résultat:



Voici une démo .

Exemple: `align-content: center` sur un flexbox vertical

CSS:

```
div#container {
  display: flex;
  flex-direction: column;
  align-items: center;
}
```

Résultat:

Voici une démo .

Exemple: Combinaison pour centrage sur flexbox horizontal

```
div#container {
  display: flex;
  flex-direction: row;
  justify-content: center;
  align-items: center;
}
```

Résultat:

Voici une démo .

Exemple: Combinaison pour centrer les deux sur la boîte flexible verticale

```
div#container {
  display: flex;
  flex-direction: column;
  justify-content: center;
  align-items: center;
}
```

Résultat:

Voici une démo .

Même hauteur sur les conteneurs imbriqués

Ce code garantit que tous les conteneurs imbriqués ont toujours la même hauteur. Cela se fait en s'assurant que tous les éléments imbriqués ont la même hauteur que le div parrent contenant.
Voir exemple de travail : https://jsfiddle.net/3wwh7ewp/

Cet effet est obtenu car les propriétés `align-items` sont définies pour `stretch` par défaut.

HTML

```
<div class="container">
  <div style="background-color: red">
    Some <br />
    data <br />
    to make<br />
    a height <br />
 </div>
  <div style="background-color: blue">
```

```
    Fewer <br />
    lines <br />
 </div>
</div>
```

CSS

```
.container {
    display: flex;
    align-items: stretch; // Default value
}
```

Note: Ne fonctionne pas sur les versions IE sous 10

Ajuster les éléments de manière optimale à leur conteneur

L'une des fonctionnalités les plus intéressantes de Flexbox est de permettre l'adaptation optimale des conteneurs à leur élément parent.

Démo en direct .

HTML:

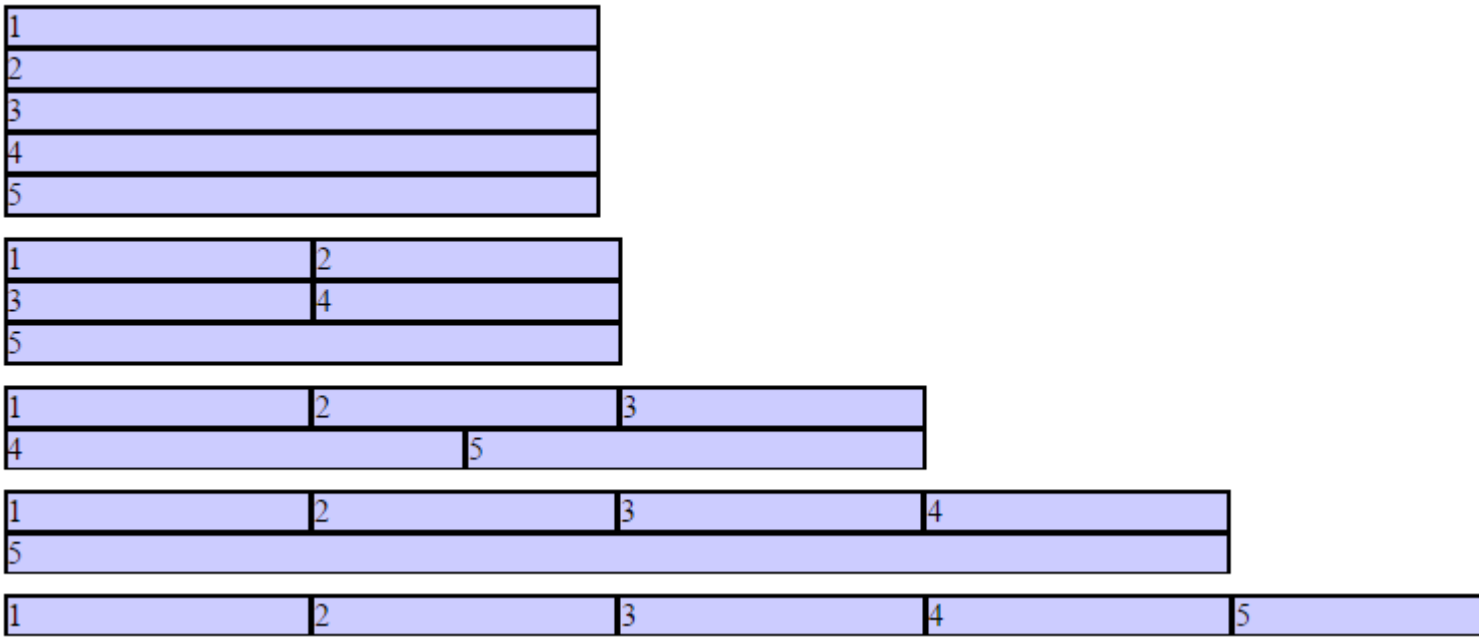
```
<div class="flex-container">
  <div class="flex-item">1</div>
  <div class="flex-item">2</div>
  <div class="flex-item">3</div>
  <div class="flex-item">4</div>
  <div class="flex-item">5</div>
</div>
```

CSS:

```
.flex-container {
    background-color: #000;
    height: 100%;
    display:flex;
    flex-direction: row;
    flex-wrap: wrap;
    justify-content: flex-start;
    align-content: stretch;
    align-items: stretch;
}

.flex-item {
    background-color: #ccf;
    margin: 0.1em;
    flex-grow: 1;
    flex-shrink: 0;
    flex-basis: 200px; /* or % could be used to ensure a specific layout */
}
```

Résultat:

Les colonnes s'adaptent à mesure que l'écran est redimensionné.



Lire Disposition de la boîte flexible (Flexbox) en ligne:
https://riptutorial.com/fr/css/topic/445/disposition-de-la-boite-flexible--flexbox-

Chapitre 20: Disposition en ligne intégrée

Examples

Barre de navigation justifiée

La barre de navigation (menu) justifiée horizontalement contient un certain nombre d'éléments à justifier. Le premier élément (à gauche) n'a pas de marge gauche dans le conteneur, le dernier élément (à droite) n'a pas de marge droite dans le conteneur. La distance entre les éléments est la même, indépendamment de la largeur de chaque élément.

HTML

```
<nav>
    <ul>
        <li>abc</li>
        <li>abcdefghijkl</li>
        <li>abcdef</li>
    </ul>
</nav>
```

CSS

```
nav {
    width: 100%;
    line-height: 1.4em;
}
ul {
    list-style: none;
    display: block;
    width: 100%;
    margin: 0;
    padding: 0;
    text-align: justify;
    margin-bottom: -1.4em;
}
ul:after {
    content: "";
    display: inline-block;
    width: 100%;
}
li {
    display: inline-block;
}
```

Remarques

- Les balises `nav` , `ul` et `li` ont été choisies pour leur signification sémantique «une liste d'éléments de navigation (menu)». D'autres balises peuvent également être utilisées bien sûr.
- Le pseudo-élément `:after` provoque une "ligne" supplémentaire dans l' `ul` et donc une hauteur supplémentaire vide de ce bloc, poussant d'autres contenus vers le bas. Ceci est résolu par la `margin-bottom` négative en `margin-bottom` , qui doit avoir la même amplitude que la `line-height` la `line-height` (mais négative).
- Si la page devient trop étroite pour que tous les éléments soient ajustés, les articles se retrouveront sur une nouvelle ligne (en partant de la droite) et seront justifiés sur cette ligne. La hauteur totale du menu augmentera au besoin.

Lire Disposition en ligne intégrée en ligne: https://riptutorial.com/fr/css/topic/3308/disposition-en-ligne-integree

Chapitre 21: Formes à un seul élément

Examples

Carré

Pour créer un carré, définissez un élément avec une largeur et une hauteur. Dans l'exemple ci-dessous, nous avons un élément avec une `width` et une `height` de 100 pixels chacune.

```
<div class="square"></div>
```

```
.square {
    width: 100px;
    height: 100px;
    background: rgb(246, 156, 85);
}
```

Triangles

Pour créer un triangle CSS, définissez un élément avec une largeur et une hauteur de 0 pixel. La forme de triangle sera formée en utilisant les propriétés de bordure. Pour un élément avec 0 hauteur et largeur, les 4 bordures (haut, droite, bas, gauche) forment chacune un triangle. Voici un élément avec 0 hauteur / largeur et 4 bordures de couleur différentes.

En définissant des bordures sur transparentes et d'autres sur une couleur, nous pouvons créer divers triangles. Par exemple, dans le triangle Haut, nous définissons la bordure inférieure sur la couleur souhaitée, puis définissons les bordures gauche et droite sur transparentes. Voici une image avec les bords gauche et droit légèrement ombrés pour montrer comment le triangle se forme.

Les dimensions du triangle peuvent être modifiées en modifiant les différentes largeurs de bordure - plus grandes, plus courtes, asymétriques, etc. Les exemples ci-dessous montrent tous un triangle de 50x50 pixels.

Triangle - Pointage vers le haut

```
<div class="triangle-up"></div>
```

```
.triangle-up {
  width: 0;
  height: 0;
  border-left: 25px solid transparent;
  border-right: 25px solid transparent;
  border-bottom: 50px solid rgb(246, 156, 85);
}
```

Triangle - Pointage vers le bas

```
<div class="triangle-down"></div>
```

```
.triangle-down {
  width: 0;
  height: 0;
  border-left: 25px solid transparent;
  border-right: 25px solid transparent;
  border-top: 50px solid rgb(246, 156, 85);
}
```

Triangle - Pointage à droite

```
<div class="triangle-right"></div>
```

```
.triangle-right {
  width: 0;
  height: 0;
  border-top: 25px solid transparent;
  border-bottom: 25px solid transparent;
  border-left: 50px solid rgb(246, 156, 85);
}
```

Triangle - Pointant Gauche

```
<div class="triangle-left"></div>
```

```
.triangle-left {
  width: 0;
  height: 0;
  border-top: 25px solid transparent;
  border-bottom: 25px solid transparent;
  border-right: 50px solid rgb(246, 156, 85);
}
```

Triangle - Pointage vers le haut / droite

```
<div class="triangle-up-right"></div>
```

```
.triangle-up-right {
  width: 0;
  height: 0;
  border-top: 50px solid rgb(246, 156, 85);
  border-left: 50px solid transparent;
}
```

Triangle - Pointage vers le haut / gauche

```
<div class="triangle-up-left"></div>
```

```
.triangle-up-left {
  width: 0;
  height: 0;
  border-top: 50px solid rgb(246, 156, 85);
  border-right: 50px solid transparent;
}
```

Triangle - Pointage vers le bas / droite

```
<div class="triangle-down-right"></div>
```

```
.triangle-down-right {
  width: 0;
  height: 0;
  border-bottom: 50px solid rgb(246, 156, 85);
  border-left: 50px solid transparent;
}
```

Triangle - Pointage vers le bas / gauche

```
<div class="triangle-down-left"></div>
```

```
.triangle-down-left {
  width: 0;
  height: 0;
  border-bottom: 50px solid rgb(246, 156, 85);
  border-right: 50px solid transparent;
```

```
}
```

Éclats

Une rafale est similaire à une étoile mais les points s'étendent moins loin du corps. Pensez à une forme de rafale comme à un carré avec des carrés supplémentaires légèrement rotatifs superposés.

Les places supplémentaires créées par la méthode `::before` et `::after` Psuedo-éléments.

8 Point Burst

Une rafale de 8 points est 2 carrés superposés. Le carré inférieur est l'élément lui-même, le carré supplémentaire est créé à l'aide du pseudo-élément `:before` . Le fond est tourné de 20 °, le carré supérieur est tourné de 135 °.

```
<div class="burst-8"></div>

.burst-8 {
  background: rgb(246, 156, 85);
  width: 40px;
  height: 40px;
  position: relative;
  text-align: center;
  -ms-transform: rotate(20deg);
  transform: rotate(20eg);
}

.burst-8::before {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  height: 40px;
  width: 40px;
  background: rgb(246, 156, 85);
  -ms-transform: rotate(135deg);
  transform: rotate(135deg);
}
```

12 point Burst

Un éclat de 12 points est composé de 3 carrés superposés. Le carré du bas est l'élément lui-même, les carrés supplémentaires sont créés à l'aide des pseudo-éléments `:before` et `:after` . Le fond est tourné de 0 °, le carré suivant est tourné de 30 ° et le haut est tourné de 60 °.

```
<div class="burst-12"></div>

.burst-12 {
  width: 40px;
  height: 40px;
  position: relative;
  text-align: center;
  background: rgb(246, 156, 85);
}

.burst-12::before, .burst-12::after {
  content: "";
  position: absolute;
  top: 0;
  left: 0;
  height: 40px;
  width: 40px;
  background: rgb(246, 156, 85);
}

.burst-12::before {
  -ms-transform: rotate(30deg);
  transform: rotate(30deg);
}

.burst-12::after {
  -ms-transform: rotate(60deg);
  transform: rotate(60deg);
}
```

Cercles et Ellipses

Cercle

Pour créer un **cercle** , définissez un élément de même `width` et `height` (un *carré*), puis définissez la propriété `border-radius` de cet élément sur `50%` .

HTML

```
<div class="circle"></div>
```

CSS

```
.circle {
   width: 50px;
   height: 50px;
   background: rgb(246, 156, 85);
   border-radius: 50%;
}
```

Ellipse

Une **ellipse** est similaire à un cercle, mais avec des valeurs différentes pour la `width` et la `height` .

HTML

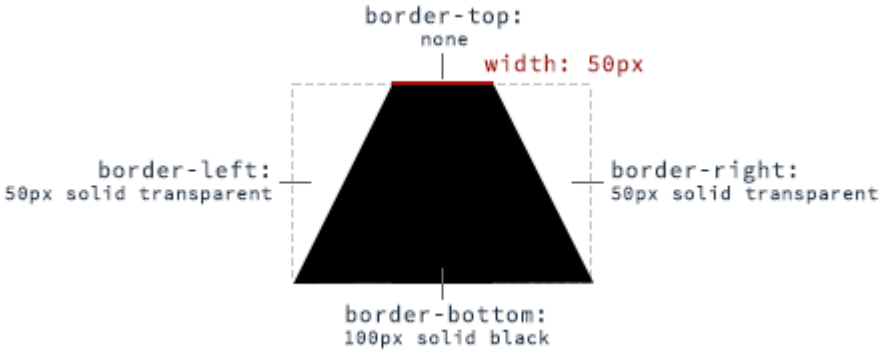
```
<div class="oval"></div>
```

CSS

```
.oval {
  width: 50px;
  height: 80px;
  background: rgb(246, 156, 85);
  border-radius: 50%;
}
```

Trapèze

Un trapèze peut être créé par un élément de bloc à hauteur nulle (hauteur de `0px`), une largeur supérieure à zéro et une bordure transparente, sauf pour un côté:



HTML:

```
<div class="trapezoid"></div>
```

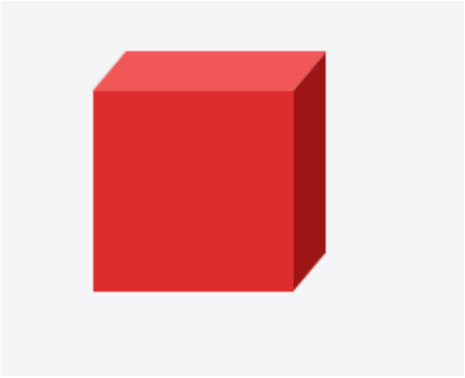
CSS:

```
.trapezoid {
    width: 50px;
    height: 0;
    border-left: 50px solid transparent;
    border-right: 50px solid transparent;
    border-bottom: 100px solid black;
}
```

En changeant les côtés de la bordure, l'orientation du trapèze peut être ajustée.

cube

Cet exemple montre comment créer un cube à l'aide des méthodes de transformation 2D `skewX()` et `skewY()` sur des pseudo-éléments.



HTML:

```
<div class="cube"></div>
```

CSS:

```
.cube {
  background: #dc2e2e;
  width: 100px;
  height: 100px;
  position: relative;
  margin: 50px;
}

.cube::before {
  content: '';
  display: inline-block;
  background: #f15757;
  width: 100px;
  height: 20px;
  transform: skewX(-40deg);
  position: absolute;
  top: -20px;
  left: 8px;
```

```
}

.cube::after {
  content: '';
  display: inline-block;
  background: #9e1515;
  width: 16px;
  height: 100px;
  transform: skewY(-50deg);
  position: absolute;
  top: -10px;
  left: 100%;
}
```

Voir la démo

Pyramide

Cet exemple montre comment créer une **pyramide à l'** aide de bordures et de méthodes de transformation 2D `skewY()` et `rotate()` sur des pseudo-éléments.

HTML:

```
<div class="pyramid"></div>
```

CSS:

```
.pyramid {
  width: 100px;
  height: 200px;
  position: relative;
  margin: 50px;
}

.pyramid::before,.pyramid::after {
  content: '';
  display: inline-block;
  width: 0;
  height: 0;
  border: 50px solid;
  position: absolute;
}

.pyramid::before {
  border-color: transparent transparent #ff5656 transparent;
```

```
  transform: scaleY(2) skewY(-40deg) rotate(45deg);
}

.pyramid::after {
  border-color: transparent transparent #d64444 transparent;
  transform: scaleY(2) skewY(40deg) rotate(-45deg);
}
```

Lire Formes à un seul élément en ligne: https://riptutorial.com/fr/css/topic/2862/formes-a-un-seul-element

Chapitre 22: Formes pour flotteurs

Syntaxe

- forme extérieure: aucune | [<basic-shape> || <forme-boîte>] | <image>
- marge de forme: <longueur> | <pourcentage>
- shape-image-threshold: <nombre>

Paramètres

Paramètre	Détails
aucun	Une valeur `none` signifie que la zone flottante (la zone utilisée pour envelopper le contenu autour d'un élément flottant) n'est pas affectée. C'est la valeur par défaut / initiale.
forme de base	Fait référence à un `inset()` parmi `inset()` , `circle()` , `ellipse()` ou `polygon()` . En utilisant l'une de ces fonctions et ses valeurs, la forme est définie.
boîte de forme	Fait référence à un `margin-box` parmi `margin-box` , `border-box` , `padding-box` et `content-box` . Lorsque seulement <shape-box> est fourni (sans <basic-shape>), cette case *est la* forme. Lorsqu'il est utilisé avec <basic-shape>, il sert de boîte de référence.
image	Lorsqu'une image est fournie en tant que valeur, la forme est calculée en fonction du canal alpha de l'image spécifiée.

Remarques

Le support du navigateur pour le module CSS Shapes est très limité à ce stade.

Il est pris en charge dans Chrome v37 + et Opera 24+ sans préfixes de navigateur / fournisseur. Safari le prend en charge à partir de la version 7.1 + mais avec le préfixe `-webkit-` .

Il n'est pas encore pris en charge dans IE, Edge et Firefox.

Examples

Forme extérieure avec forme de base - cercle ()

Avec la propriété CSS `shape-outside` la `shape-outside` , vous pouvez définir des valeurs de forme pour la zone flottante afin que le contenu en ligne contourne la forme plutôt que la zone du flottant.

CSS

```
img:nth-of-type(1) {
  shape-outside: circle(80px at 50% 50%);
  float: left;
  width: 200px;
}
img:nth-of-type(2) {
  shape-outside: circle(80px at 50% 50%);
  float: right;
  width: 200px;
}
p {
  text-align: center;
  line-height: 30px; /* purely for demo */
}
```

HTML

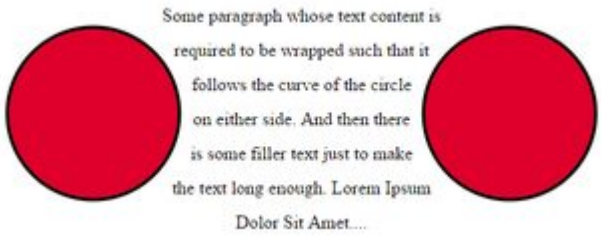
```
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<p>Some paragraph whose text content is required to be wrapped such that it follows the curve
of the circle on either side. And then there is some filler text just to make the text long
enough. Lorem Ipsum Dolor Sit Amet....</p>
```

Dans l'exemple ci-dessus, les images sont en fait des images carrées et, lorsque le texte est placé sans la propriété `shape-outside` , il ne circulera pas autour du cercle. Il circulera autour de la boîte contenant de l'image uniquement. Avec `shape-outside` la zone float est redéfinie comme un *cercle* et le contenu est fait pour circuler autour de ce *cercle imaginaire* créé en utilisant `shape-outside` .

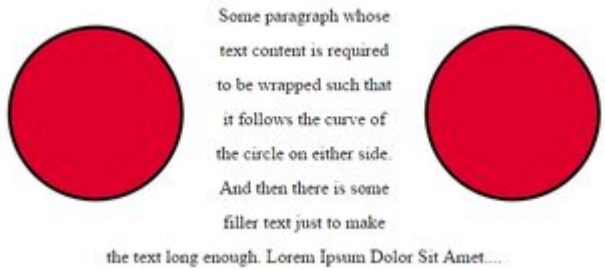
Le *cercle imaginaire* utilisé pour redéfinir la zone flottante est un cercle de rayon de 80px tiré du centre de la zone de référence de l'image.

Vous trouverez ci-dessous quelques captures d'écran pour illustrer comment le contenu serait utilisé lorsque la `shape-outside` est utilisée et quand elle n'est pas utilisée.

Sortie avec `shape-outside`



Sortie sans `shape-outside`



Marge de forme

La `shape-margin` propriété CSS ajoute une *marge* de `shape-outside` à l' `shape-outside` .

CSS

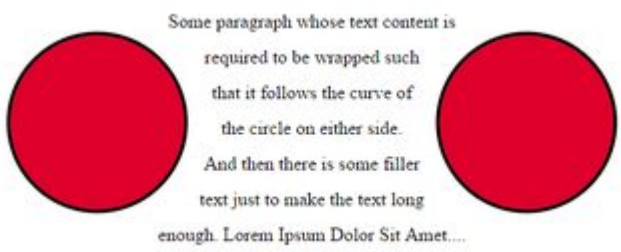
```
img:nth-of-type(1) {
  shape-outside: circle(80px at 50% 50%);
  shape-margin: 10px;
  float: left;
  width: 200px;
}
img:nth-of-type(2) {
  shape-outside: circle(80px at 50% 50%);
  shape-margin: 10px;
  float: right;
  width: 200px;
}
p {
  text-align: center;
  line-height: 30px; /* purely for demo */
}
```

HTML

```
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<img src="http://images.clipartpanda.com/circle-clip-art-circlergb.jpg">
<p>Some paragraph whose text content is required to be wrapped such that it follows the curve
of the circle on either side. And then there is some filler text just to make the text long
enough. Lorem Ipsum Dolor Sit Amet....</p>
```

Dans cet exemple, une marge de 10px est ajoutée autour de la **forme en** utilisant `shape-margin` . Cela crée un peu plus d'espace entre le *cercle imaginaire* qui définit la zone flottante et le contenu réel qui circule.

Sortie:



Lire Formes pour flotteurs en ligne: https://riptutorial.com/fr/css/topic/2034/formes-pour-flotteurs

Chapitre 23: Fragmentation

Syntaxe

- page-break-after: auto | toujours | éviter | à gauche | droit | initiale | hériter;
- page-break-before: auto | toujours | éviter | à gauche | droit | initiale | hériter;
- page-break-inside: auto | éviter | initiale | hériter;

Paramètres

Valeur	La description
auto	Défaut. Saut de page automatique
toujours	Toujours insérer un saut de page
éviter	Évitez le saut de page (si possible)
la gauche	Insérer des sauts de page afin que la page suivante soit formatée en tant que page de gauche
droite	Insérer des sauts de page afin que la page suivante soit formatée comme une page de droite
initiale	Définit cette propriété sur sa valeur par défaut.
hériter	Hérite cette propriété de son élément parent.

Remarques

Il n'y a pas de propriété de saut de page dans CSS. Seules les 3 propriétés (**page-break-before** , **page-break-after** , **page-break-inside**).

Connexes: `orphans` , `widows` .

Examples

Médias imprimer page-pause

```
@media print {
  p {
    page-break-inside: avoid;
  }
  h1 {
    page-break-before: always;
```

```
  }
  h2 {
    page-break-after: avoid;
  }
}
```

Ce code fait 3 choses:

- il empêche un saut de page à l'intérieur de balises p, ce qui signifie qu'un paragraphe ne sera jamais divisé en deux pages, si possible.
- il force un saut de page avant dans tous les en-têtes h1, ce qui signifie qu'avant chaque occurrence de h1, il y aura un saut de page.
- il empêche les sauts de page juste après un h2

Lire Fragmentation en ligne: https://riptutorial.com/fr/css/topic/4316/fragmentation

Chapitre 24: Frontière

Syntaxe

- **frontière**
- border: border-width border-style border-color | initiale | hériter;
- border-top: border-width style de bordure border-color | initiale | hériter;
- border-bottom: bordure-largeur style-bordure bordure-couleur | initiale | hériter;
- border-left: border-style border-style border-color | initiale | hériter;
- border-right: border-width style de bordure border-color | initiale | hériter;
- **style de bordure**
- style de bordure: 1-4 aucune | caché | pointillé | en pointillés | solide | double | rainure | crête | encart | début | initiale | hériter;
- **rayon frontière**
- rayon frontière: 1-4 longueur | % / 1-4 longueur | % | initiale | hériter;
- border-top-left-radius: longueur | % [longueur | %] | initiale | hériter;
- border-top-right-radius: longueur | % [longueur | %] | initiale | hériter;
- border-bottom-left-radius: longueur | % [longueur | %] | initiale | hériter;
- border-bottom-right-radius: longueur | % [longueur | %] | initiale | hériter;
- **image-frontière**
- border-image: border-image-source border-image-slice [bordure-image-largeur [bordure-image-sortie]] border-image-repeat
- border-image-source: aucune | image;
- border-image-slice: 1-4 nombre | pourcentage [remplir]
- border-image-repeat: 1-2 stretch | répéter | ronde | espace
- **effondrement des frontières**
- effondrement des frontières: séparé | effondrement initiale | hériter

Remarques

Propriétés connexes:

- frontière
- fond de frontière
- border-bottom-colour
- frontière-bas-rayon gauche
- frontière-bas-droit-rayon
- border-bottom-style
- border-bottom-width
- couleur de la bordure
- image-frontière
- border-image-outset
- border-image-repeat
- border-image-slice
- border-image-source
- border-image-width
- frontière gauche
- border-left-color
- style frontière-gauche
- border-left-width
- rayon frontière
- frontière droite
- border-right-color
- style frontière
- border-right-width
- style de bordure
- haut de la frontière
- bord-haut-couleur

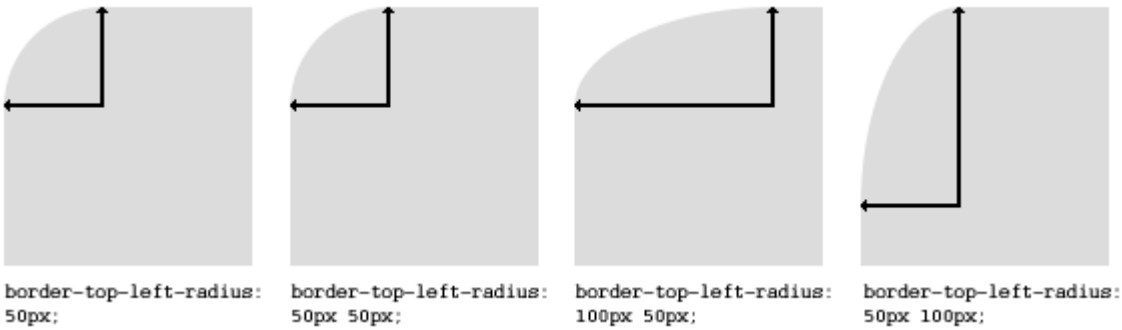
- frontière-haut-rayon gauche
- frontière-haut-droit-rayon
- style frontière
- border-top-width
- largeur de la bordure

Examples

rayon frontière

La propriété border-radius vous permet de modifier la forme du modèle de boîte de base.

Chaque coin d'un élément peut avoir jusqu'à deux valeurs, pour le rayon vertical et horizontal de ce coin (pour un maximum de 8 valeurs).



Le premier ensemble de valeurs définit le rayon horizontal. Le deuxième ensemble de valeurs facultatif, précédé d'un '/', définit le rayon vertical. Si un seul jeu de valeurs est fourni, il est utilisé pour le rayon vertical et le rayon horizontal.

```
border-radius: 10px 5% / 20px 25em 30px 35em;
```

Le `10px` est le rayon horizontal du haut à gauche et du bas à droite. Et le `5%` est le rayon horizontal du haut à droite et du bas à gauche. Les quatre autres valeurs après '/' sont les rayons verticaux pour le haut à gauche, le haut à droite, le bas à droite et le bas à gauche.

Comme avec de nombreuses propriétés CSS, les raccourcis peuvent être utilisés pour toutes les valeurs possibles. Vous pouvez donc spécifier de 1 à 8 valeurs. Le raccourci suivant vous permet de définir le rayon horizontal et vertical de chaque coin sur la même valeur:

HTML:

```
<div class='box'></div>
```

CSS:

```
.box {
    width: 250px;
    height: 250px;
    background-color: black;
    border-radius: 10px;
}
```

Le rayon de bordure est le plus souvent utilisé pour convertir des éléments de boîte en cercles. En définissant le rayon de la bordure sur la moitié de la longueur d'un élément carré, un élément circulaire est créé:

```
.circle {
    width: 200px;
    height: 200px;
    border-radius: 100px;
}
```

Comme le paramètre border-radius accepte des pourcentages, il est courant d'utiliser 50% pour éviter de calculer manuellement la valeur border-radius:

```
.circle {
    width: 150px;
    height: 150px;
    border-radius: 50%;
}
```

Si les propriétés de largeur et de hauteur ne sont pas égales, la forme résultante sera un ovale plutôt qu'un cercle.

Exemple de bordure-rayon spécifique au navigateur:

```
  -webkit-border-top-right-radius: 4px;
  -webkit-border-bottom-right-radius: 4px;
  -webkit-border-bottom-left-radius: 0;
  -webkit-border-top-left-radius: 0;
  -moz-border-radius-topright: 4px;
  -moz-border-radius-bottomright: 4px;
  -moz-border-radius-bottomleft: 0;
  -moz-border-radius-topleft: 0;
  border-top-right-radius: 4px;
  border-bottom-right-radius: 4px;
  border-bottom-left-radius: 0;
  border-top-left-radius: 0;
```

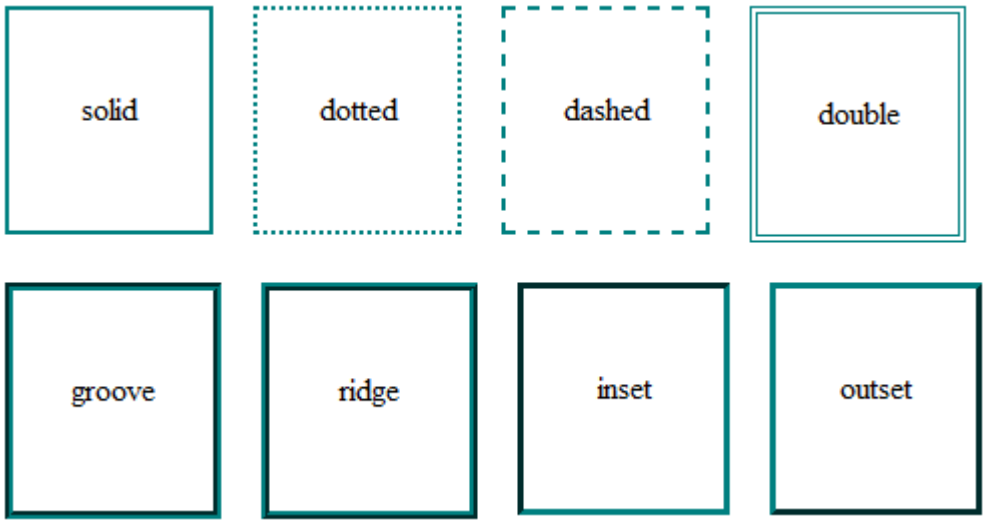
style de bordure

La propriété `border-style` définit le style de la bordure d'un élément. Cette propriété peut avoir de une à quatre valeurs (pour chaque côté de l'élément une valeur.)

Exemples:

```
border-style: dotted;
```

```
border-style: dotted solid double dashed;
```



`border-style` peut également avoir les valeurs `none` et `hidden` . Ils ont le même effet, à l'exception `hidden` travaux `hidden` pour la résolution des conflits de bordure pour les éléments `<table>` . Dans une `<table>` avec plusieurs bordures, `none` possède la priorité la plus faible (signifiant un conflit, la bordure le montrerait) et la priorité la plus élevée est `hidden` (ce qui signifie que la bordure ne s'affiche pas dans un conflit).

frontière (sténographie)

Dans la plupart des cas, vous souhaitez définir plusieurs propriétés de bordure (`border-width` , `border-style` et `border-color`) pour tous les côtés d'un élément.

Au lieu d'écrire:

```
border-width: 1px;
border-style: solid;
border-color: #000;
```

Vous pouvez simplement écrire:

```
border: 1px solid #000;
```

Ces raccourcis sont également disponibles pour chaque côté d'un élément: `border-top` , `border-left` , `border-right` et `border-bottom` . Donc, vous pouvez faire:

```
border-top: 2px double #aaaaaa;
```

image-frontière

Avec la propriété `border-image` , vous avez la possibilité de définir une image à utiliser à la place des styles de bordure normaux.

Une `border-image` consiste essentiellement en un

- `border-image-source` : le chemin de l'image à utiliser
- `border-image-slice` : spécifie le décalage utilisé pour diviser l'image en **neuf régions** (quatre **coins** , quatre **arêtes** et un **centre**)
- `border-image-repeat` : Spécifie comment les images pour les côtés et le milieu de l'image de la bordure sont mises à l'échelle.

Prenons l'exemple suivant: borderas.png est une image de 90x90 pixels:

```
border-image: url("border.png") 30 stretch;
```

L'image sera divisée en neuf régions de 30x30 pixels. Les bords seront utilisés comme coins de la bordure tandis que le côté sera utilisé entre les deux. Si l'élément est plus haut / plus large que 30px, cette partie de l'image sera **étirée** . La partie centrale de l'image est par défaut transparente.

border- [left | right | top | bottom]

La propriété `border-[left|right|top|bottom]` est utilisée pour ajouter une bordure à un côté spécifique d'un élément.

Par exemple, si vous souhaitez ajouter une bordure à la gauche d'un élément, vous pouvez faire:

```
#element {
    border-left: 1px solid black;
}
```

effondrement des frontières

La propriété `border-collapse` s'applique uniquement aux `table` (et aux éléments affichés sous la forme `display: table` ou `inline-table`) et définit si les bordures du tableau sont réduites en une seule bordure ou détachées comme dans le code HTML standard.

```
table {
  border-collapse: separate; /* default */
  border-spacing: 2px; /* Only works if border-collapse is separate */
}
```

Voir aussi Tableaux - Entrée de la documentation border-collapse

Frontières multiples

Utiliser le contour:

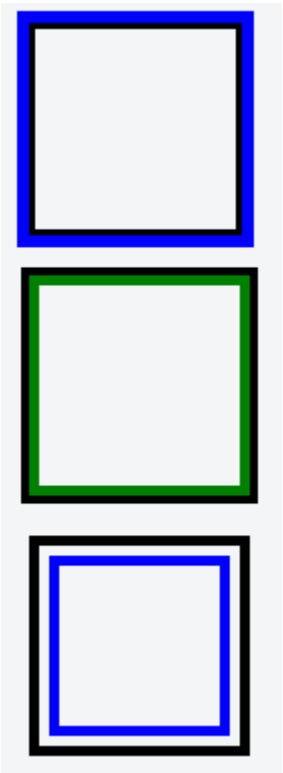
```
.div1{
  border: 3px solid black;
  outline: 6px solid blue;
  width: 100px;
  height: 100px;
  margin: 20px;
}
```

En utilisant box-shadow:

```
.div2{
  border: 5px solid green;
  box-shadow: 0px 0px 0px 4px #000;
  width: 100px;
  height: 100px;
  margin: 20px;
}
```

Utiliser un pseudo-élément:

```
.div3 {
  position: relative;
  border: 5px solid #000;
  width: 100px;
  height: 100px;
  margin: 20px;
}
.div3:before {
  content: " ";
  position: absolute;
  border: 5px solid blue;
  z-index: -1;
  top: 5px;
  left: 5px;
  right: 5px;
  bottom: 5px;
}
```



http://jsfiddle.net/MadalinaTn/bvqpcohm/2/

Créer une bordure multicolore en utilisant border-image

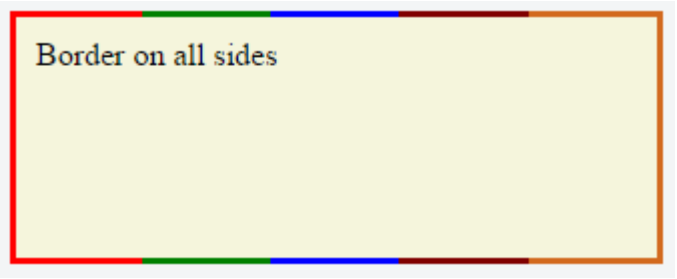
CSS

```
.bordered {
  border-image: linear-gradient(to right, red 20%, green 20%, green 40%, blue 40%, blue 60%,
maroon 60%, maroon 80%, chocolate 80%); /* gradient with required colors */
  border-image-slice: 1;
}
```

HTML

```
<div class='bordered'>Border on all sides</div>
```

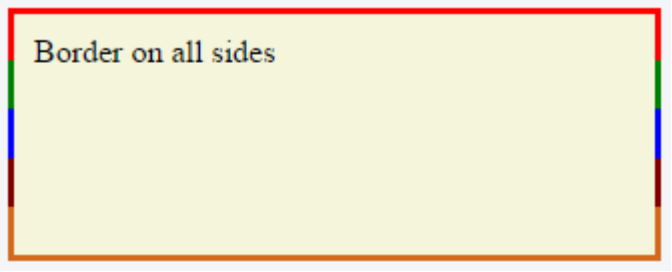
L'exemple ci-dessus produirait une bordure composée de 5 couleurs différentes. Les couleurs sont définies par un `linear-gradient` (vous pouvez trouver plus d'informations sur les dégradés dans les documents). Vous pouvez trouver plus d'informations sur la propriété `border-image-slice` dans l' exemple `border-image` dans la même page.



(*Remarque: des propriétés supplémentaires ont été ajoutées à l'élément à des fins de présentation.*)

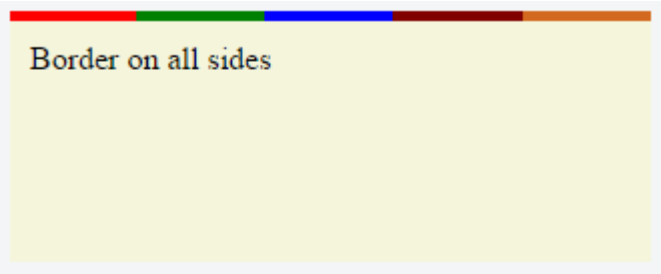
Vous auriez remarqué que la bordure gauche n'a qu'une seule couleur (la couleur de départ du dégradé) alors que la bordure droite ne comporte qu'une seule couleur (la couleur de fin du dégradé). Cela est dû à la façon dont la propriété image frontière fonctionne. C'est comme si le dégradé était appliqué à la boîte entière et que les couleurs étaient masquées à partir des zones de remplissage et de contenu, donnant ainsi l'impression que seule la bordure est dégradée.

La ou les bordures d'une seule couleur dépendent de la définition du dégradé. Si le dégradé est un dégradé de `to right` , la bordure gauche correspond à la couleur de début du dégradé et la bordure droite correspond à la couleur de fin. Si c'était un dégradé du `to bottom` la bordure supérieure serait la couleur de départ du dégradé et la bordure du bas serait la couleur de fin. Ci - dessous, la sortie d'un `to bottom` en `to bottom` 5 gradient de couleur.



Si la bordure est requise uniquement sur des côtés spécifiques de l'élément, la propriété `border-width` peut être utilisée comme avec toute autre bordure normale. Par exemple, l'ajout du code ci-dessous produirait une bordure uniquement en haut de l'élément.

```
border-width: 5px 0px 0px 0px;
```



Notez que tout élément qui possède `border-image` propriété `border-image` **ne respectera pas le** `border-radius` (c'est-à-dire que la bordure ne sera pas courbe). Ceci est basé sur la déclaration ci-dessous dans la spécification:

> Les arrière-plans d'une boîte, mais pas sa bordure-image, sont découpés dans la courbe appropriée (comme déterminé par "background-clip").

Lire Frontière en ligne: https://riptutorial.com/fr/css/topic/2160/frontiere

Chapitre 25: Grandes lignes

Syntaxe

- contour: contour contour couleur | initiale | hériter;
- largeur du contour: moyen | mince | épais | longueur | initiale | hériter;
- style de contour: aucun | caché | pointillé | en pointillés | solide | double | rainure | crête | encart | début | initiale | hériter;

Paramètres

Paramètre	Détails
pointé	contour en pointillé
en pointillés	contour en pointillés
solide	contour solide
double	double contour
rainure	Contour 3D rainuré, dépend de la valeur de la couleur du contour
crête	Contour 3D strié, dépend de la valeur de la couleur du contour
encart	Contour en 3D, dépend de la valeur de la couleur du contour
début	Contour des 3D, dépend de la valeur de la couleur du contour
aucun	pas de contour
caché	contour caché

Remarques

`outline` sont maintenant décrites dans Basic UI , un module CSS de niveau 3 (il était déjà décrit dans REC CSS2.1)

La propriété Outline est définie par défaut dans les navigateurs pour les éléments pouvant être `:focus` dans `:focus` état de `:focus` .
Il ne devrait pas être supprimé, voir http://outlinenone.com qui indique:

> **Que fait la propriété contour?**
>
> Il fournit un retour visuel pour les liens qui ont "focus" lors de la navigation dans un

> document Web à l'aide de la touche TAB (ou équivalente). Ceci est particulièrement utile pour les personnes qui ne peuvent pas utiliser une souris ou qui ont une déficience visuelle. Si vous supprimez le contour, vous rendrez votre site inaccessible pour ces personnes. (…)

Exemples intéressants sur le débordement de pile:

- Comment supprimer le contour de bordure sur un élément de texte en entrée
- Comment supprimer le contour pointillé de Firefox sur les boutons ainsi que les liens?

Examples

Vue d'ensemble

Le contour est une ligne qui contourne l'élément, en dehors de la bordure. Contrairement aux `border` , les contours ne prennent aucune place dans le modèle de boîte. L'ajout d'un contour à un élément n'affecte donc pas la position de l'élément ou d'autres éléments.

De plus, les contours peuvent être non rectangulaires dans certains navigateurs. Cela peut se produire si le `outline` est appliqué à un élément `span` contenant du texte avec des propriétés de `font-size` différentes. Contrairement aux bordures, les contours *ne peuvent pas* avoir de coins arrondis.

Les parties essentielles du `outline` sont la `outline-color` `outline-style` , le `outline-style` `outline-width` .

La définition d'un contour est équivalente à la définition d'une bordure:

> Un contour est une ligne autour d'un élément. Il est affiché autour de la marge de l'élément. Cependant, il est différent de la propriété frontière.

```
outline: 1px solid black;
```

style de contour

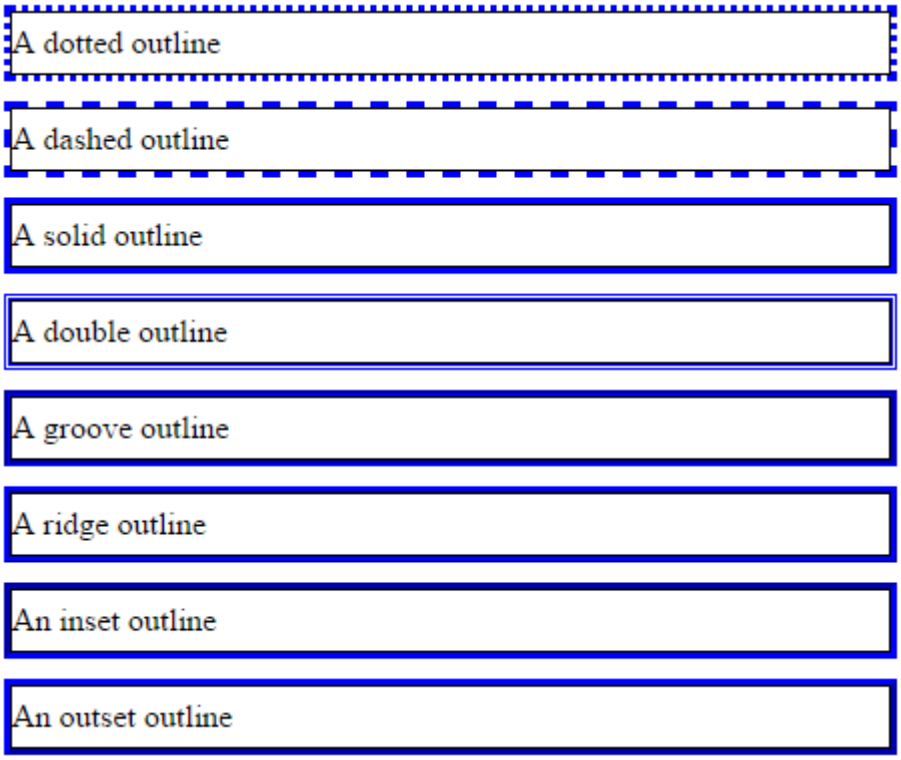
La propriété de `outline-style` est utilisée pour définir le style du contour d'un élément.

```
p {
  border: 1px solid black;
  outline-color:blue;
  line-height:30px;
}
.p1{
  outline-style: dotted;
}
.p2{
  outline-style: dashed;
}
.p3{
  outline-style: solid;
```

```
}
.p4{
  outline-style: double;
}
.p5{
  outline-style: groove;
}
.p6{
  outline-style: ridge;
}
.p7{
  outline-style: inset;
}
.p8{
  outline-style: outset;
}
```

HTML

```
<p class="p1">A dotted outline</p>
<p class="p2">A dashed outline</p>
<p class="p3">A solid outline</p>
<p class="p4">A double outline</p>
<p class="p5">A groove outline</p>
<p class="p6">A ridge outline</p>
<p class="p7">An inset outline</p>
<p class="p8">An outset outline</p>
```



Lire Grandes lignes en ligne: https://riptutorial.com/fr/css/topic/4258/grandes-lignes

Chapitre 26: Héritage

Syntaxe

- *propriété:* hériter;

Examples

Héritage automatique

Héritage un mécanisme fondamental de CSS par lequel les valeurs calculées de certaines propriétés d'un élément sont appliquées à ses enfants. Ceci est particulièrement utile lorsque vous souhaitez définir un style global pour vos éléments plutôt que de définir ces propriétés sur chacun des éléments de votre balisage.

Les propriétés communes qui sont automatiquement héritées sont: `font` , `color` , `text-align` , `line-height` .

Supposons la feuille de style suivante:

```
#myContainer {
  color: red;
  padding: 5px;
}
```

Cela appliquera la `color: red` non seulement à l'élément `<div>` mais aussi aux éléments `<h3>` et `<p>` . Cependant, en raison de la nature du `padding` sa valeur ***ne*** sera ***pas*** héritée de ces éléments.

```
<div id="myContainer">
  <h3>Some header</h3>
  <p>Some paragraph</p>
</div>
```

Héritage forcé

Certaines propriétés ne sont pas automatiquement héritées d'un élément jusqu'à ses enfants. En effet, ces propriétés doivent généralement être uniques à l'élément (ou à la sélection d'éléments) auquel la propriété est appliquée. Les propriétés communes sont la `margin` , le `padding` , l' `background` - `background` , l' `display` , etc.

Cependant, l'héritage est parfois souhaité de toute façon. Pour ce faire , nous pouvons appliquer la `inherit` valeur à la propriété qui doit être héritée. La `inherit` valeur peut être appied à *une* propriété CSS et *tout* élément HTML.

Supposons la feuille de style suivante:

```
#myContainer {
  color: red;
  padding: 5px;
}
#myContainer p {
  padding: inherit;
}
```

Cela appliquera la `color: red` aux deux éléments `<h3>` et `<p>` raison de la nature d'héritage de la propriété `color` . Toutefois, l'élément `<p>` héritera également de la valeur de `padding` de son parent, car cela a été spécifié.

```
<div id="myContainer">
  <h3>Some header</h3>
  <p>Some paragraph</p>
</div>
```

Lire Héritage en ligne: https://riptutorial.com/fr/css/topic/3586/heritage

Chapitre 27: Internet Explorer Hacks

Remarques

Ces «hacks» peuvent être utilisés pour cibler un navigateur / client spécifique. Cela peut être utilisé pour contourner les différences de rendu du navigateur en appliquant des styles dans l'un des wrappers répertoriés ci-dessus.

Examples

Mode Contraste élevé dans Internet Explorer 10 et supérieur

Dans Internet Explorer 10+ et Edge, Microsoft fournit le sélecteur de support `-ms-high-contrast` pour exposer le paramètre «Contraste élevé» du navigateur, ce qui permet au programmeur d'ajuster les styles de leur site en conséquence.

Le sélecteur `-ms-high-contrast` a 3 états: `active` , `black-on-white` et `white-on-black` . Dans IE10 +, il n'y avait `none` état mais cela n'est plus pris en charge dans Edge.

Exemples

```
@media screen and (-ms-high-contrast: active), (-ms-high-contrast: black-on-white) {
   .header{
      background: #fff;
      color: #000;
   }
}
```

Cela changera l'arrière-plan de l'en-tête en blanc et la couleur du texte en noir lorsque le mode de contraste élevé est actif *et* qu'il est en mode `black-on-white` .

```
@media screen and (-ms-high-contrast: white-on-black) {
   .header{
      background: #000;
      color: #fff;
   }
}
```

Semblable au premier exemple, mais il sélectionne spécifiquement l'état `white-on-black` uniquement et inverse les couleurs de l'en-tête sur un arrière-plan noir avec du texte blanc.

Plus d'information:

Documentation Microsoft sur `-ms-high-contrast`

Internet Explorer 6 et Internet Explorer 7 uniquement

Pour cibler Internet Explorer 6 et Internet Explorer 7, démarrez vos propriétés avec `*` :

```
.hide-on-ie6-and-ie7 {
    *display : none; // This line is processed only on IE6 and IE7
}
```

Les préfixes non alphanumériques (autres que les tirets et les traits de soulignement) sont ignorés dans IE6 et IE7. Ce hack fonctionne donc pour toute `property: value` paire `property: value` préfixe.

Internet Explorer 8 uniquement

Pour cibler Internet Explorer 8, placez vos sélecteurs dans `@media \0 screen { }` :

```
@media \0 screen {
    .hide-on-ie8 {
        display : none;
    }
}
```

Tout entre `@media \0 screen { }` est traité uniquement par I

Ajout de la prise en charge du bloc en ligne à IE6 et IE7

```
display: inline-block;
```

La propriété d' `display` ayant la valeur `inline-block` n'est pas prise en charge par Internet Explorer 6 et 7. Une solution consiste à:

```
zoom: 1;
*display: inline;
```

La propriété `zoom` déclenche la fonctionnalité `hasLayout` des éléments et n'est disponible que dans Internet Explorer. L' `*display` s'assure que la propriété non valide s'exécute uniquement sur les navigateurs affectés. Les autres navigateurs ignoreront simplement la règle.

Lire Internet Explorer Hacks en ligne: https://riptutorial.com/fr/css/topic/5056/internet-explorer-hacks

Chapitre 28: la grille

Introduction

La disposition de grille est un nouveau et puissant système de mise en page CSS qui permet de diviser facilement un contenu de page Web en lignes et en colonnes.

Remarques

Le niveau 1 du module CSS Grid Layout est, depuis le 9 septembre 2016, une recommandation candidate W3C. Il est considéré comme étant en phase de test (https://www.w3.org/Style/CSS/current-work) .

À compter du 3 juillet 2017, les navigateurs Internet Explorer 10 et 11 et Edge de Microsoft ne prennent en charge qu'une version antérieure de la spécification utilisant un préfixe de fournisseur.

Examples

Exemple de base

Propriété	Valeurs possibles
afficher	grille / grille en ligne

La grille CSS est définie comme une propriété d'affichage. Elle s'applique uniquement à un élément parent et à ses enfants immédiats.

Considérez le balisage suivant:

```
<section class="container">
  <div class="item1">item1</div>
  <div class="item2">item2</div>
  <div class="item3">item3</div>
  <div class="item4">item4</div>
</section>
```

Le moyen le plus simple de définir la structure de balisage ci-dessus en tant que grille consiste à définir simplement sa propriété `display` sur `grid` :

```
.container {
  display: grid;
}
```

Cependant, cela entraînera invariablement tous les éléments enfants à s'effondrer les uns sur les autres. C'est parce que les enfants ne savent pas actuellement comment se positionner dans la

grille. Mais on peut leur dire explicitement.

Nous devons d'abord indiquer à l'élément de grille `.container` le nombre de lignes et de colonnes qui constitueront sa structure et nous pouvons le faire en utilisant les propriétés `grid-columns` et `grid-rows` (notez la pluralisation):

```
.container {
  display: grid;
  grid-columns: 50px 50px 50px;
  grid-rows: 50px 50px;
}
```

Cependant, cela ne nous aide toujours pas beaucoup car nous devons donner un ordre à chaque élément enfant. Nous pouvons le faire en spécifiant les valeurs de `grid-row` `grid-column` et `grid-column` qui lui indiqueront où il se trouve dans la grille:

```
.container .item1 {
  grid-column: 1;
  grid-row: 1;
}
.container .item2 {
  grid-column: 2;
  grid-row: 1;
}
.container .item3 {
  grid-column: 1;
  grid-row: 2;
}
.container .item4 {
  grid-column: 2;
  grid-row: 2;
}
```

En attribuant à chaque élément une valeur de colonne et de ligne, il identifie l'ordre des articles dans le conteneur.

Voir un exemple de travail sur JSFiddle . Vous aurez besoin de voir ce dans IE10, IE11 ou bord pour que cela fonctionne car ce sont actuellement les seuls navigateurs supportant la mise en page Grille (avec préfixe fournisseur `-ms-`) ou d' activer un drapeau dans Chrome, Opera et Firefox selon caniuse pour pour tester avec eux.

Lire la grille en ligne: https://riptutorial.com/fr/css/topic/2152/la-grille

Chapitre 29: Le modèle de boîte

Syntaxe

- box-sizing: *paramètre* ;

Paramètres

Paramètre	Détail
`content-box`	La largeur et la hauteur de l'élément ne comprennent que la zone de contenu.
`padding-box`	La largeur et la hauteur de l'élément incluent le contenu et le remplissage.
`border-box`	La largeur et la hauteur de l'élément incluent le contenu, le remplissage et la bordure.
`initial`	Définit le modèle de boîte à son état par défaut.
`inherit`	Hérite le modèle de boîte de l'élément parent.

Remarques

À propos de la boîte de rembourrage

Cette valeur n'a été implémentée que par **Firefox** et ne doit donc pas être utilisée. Il a été supprimé dans Firefox version 50.0.

Examples

Quel est le modèle de boîte?

Les bords

Le navigateur crée un rectangle pour chaque élément du document HTML. Le modèle de boîte décrit comment le remplissage, la bordure et la marge sont ajoutés au contenu pour créer ce rectangle.

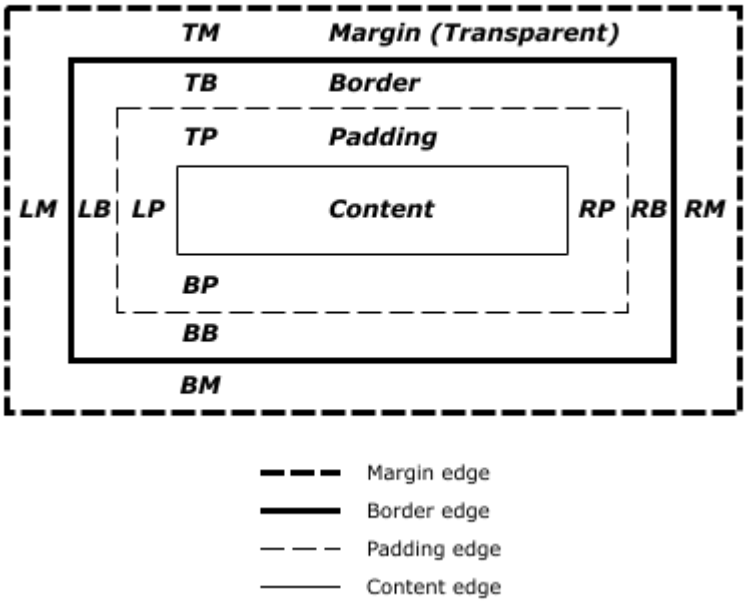


Diagramme de la version de travail CSS2.2

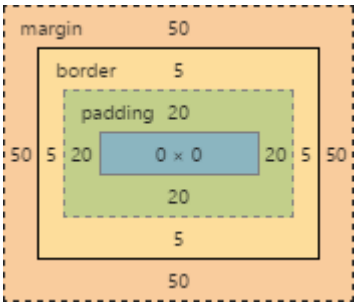
Le périmètre de chacune des quatre zones est appelé un *bord* . Chaque arête définit une *boîte.*

- Le rectangle le plus interne est la zone de **contenu** . La largeur et la hauteur de ceci dépendent du contenu rendu de l'élément (texte, images et tout élément enfant qu'il peut avoir).
- Vient ensuite la zone de **remplissage** , définie par la propriété de `padding` . Si aucune largeur de `padding` n'est définie, le bord de remplissage est égal au bord du contenu.
- Ensuite, nous avons la **boîte de bordure** , telle que définie par la propriété `border` . S'il n'y a pas de largeur de `border` définie, le bord de la bordure est égal au bord de remplissage.
- Le rectangle le plus à l'extérieur est la zone de **marge** , définie par la propriété `margin` . S'il n'y a pas de largeur de `margin` définie, le bord de la marge est égal au bord de la bordure.

Exemple

```
div {
    border: 5px solid red;
    margin: 50px;
    padding: 20px;
}
```

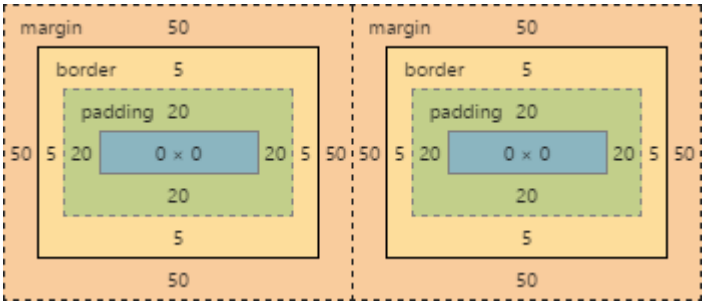
Ce style CSS classe tous les éléments `div` pour avoir une bordure supérieure, droite, inférieure et gauche de `5px` de largeur; une marge supérieure, droite, inférieure et gauche de `50px` ; et un haut, droit, bas et gauche rembourrage de `20px` . En ignorant le contenu, notre boîte générée ressemblera à ceci:



Capture d'écran du panneau Styles d'élément de Google Chrome

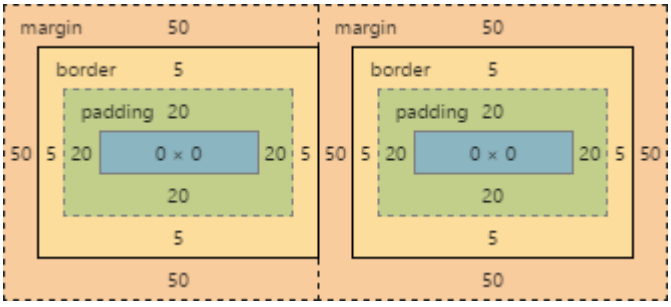
- Comme il n'y a pas de contenu, la région de contenu (la boîte bleue au milieu) n'a pas de hauteur ou de largeur (0px par 0px).
- La zone de remplissage par défaut a la même taille que la zone de contenu, plus la largeur de 20 pixels sur les quatre bords que nous définissons ci-dessus avec la propriété de `padding` (40 pixels par 40 pixels).
- La zone de bordure a la même taille que la boîte de remplissage, plus la largeur de 5 pixels que nous définissons ci-dessus avec la propriété de `border` (50 pixels par 50 pixels).
- Enfin, la zone de marge a la même taille que la zone de bordure, plus la largeur de 50px que nous définissons ci-dessus avec la propriété `margin` (ce qui donne à notre élément une taille totale de 150x150x).

Maintenant, donnons à notre élément un frère avec le même style. Le navigateur examine le modèle de boîte des deux éléments pour déterminer où, par rapport au contenu de l'élément précédent, le nouvel élément doit être positionné:



Le contenu de chacun des éléments est séparé par un espace de 150 pixels, mais les boîtes des deux éléments se touchent.

Si nous modifions ensuite notre premier élément pour ne plus avoir de marge droite, le bord de la marge droite sera dans la même position que le bord droit de la bordure et nos deux éléments ressembleront à ceci:



taille de boîte

Le modèle de boîte par défaut (`content-box`) peut être contre-intuitif, car la `width` / `height` d'un élément ne représentera pas sa largeur ou sa hauteur à l'écran dès que vous `padding` `border` styles de `padding` et de `border` à l'élément.

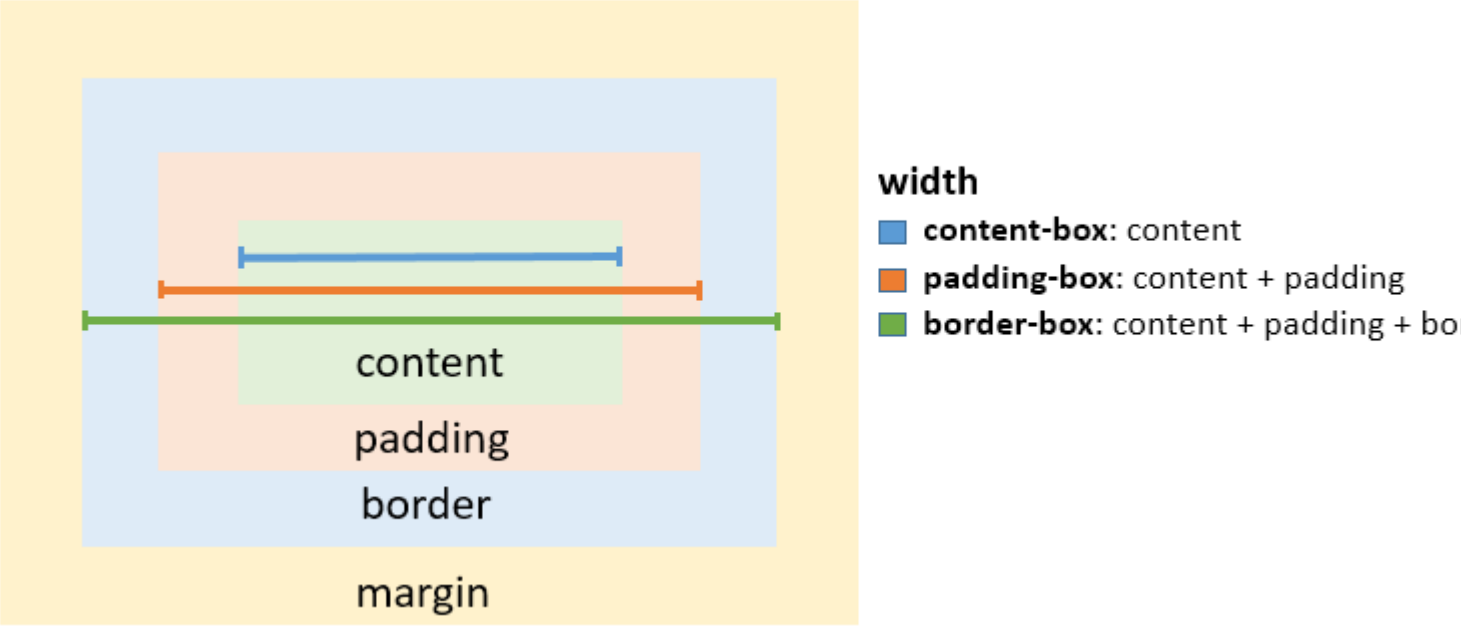
L'exemple suivant illustre ce problème potentiel avec `content-box` :

```
textarea {
    width: 100%;
    padding: 3px;
    box-sizing: content-box; /* default value */
}
```

Puisque le remplissage sera ajouté à la largeur de la zone de texte, l'élément résultant est une zone de texte plus large que 100%.

Heureusement, CSS nous permet de changer le modèle de boîte avec la propriété de `box-sizing` pour un élément. Il existe trois valeurs différentes pour la propriété disponible:

- `content-box` : Le modèle de boîte commune - la largeur et la hauteur ne comprennent que le contenu, pas le remplissage ni la bordure
- `padding-box` : La largeur et la hauteur incluent le contenu et le remplissage, mais pas la bordure
- `border-box` : La largeur et la hauteur incluent le contenu, le remplissage ainsi que la bordure



Pour résoudre le problème de `textarea` ci-dessus, vous pouvez simplement modifier la propriété de `box-sizing` en `padding-box` ou `border-box` . `border-box` est le plus couramment utilisé.

```
textarea {
```

```
    width: 100%;
    padding: 3px;
    box-sizing: border-box;
}
```

Pour appliquer un modèle de boîte spécifique à chaque élément de la page, utilisez l'extrait de code suivant:

```
html {
    box-sizing: border-box;
}

*, *:before, *:after {
    box-sizing: inherit;
}
```

Dans cette `box-sizing:border-box;` codage `box-sizing:border-box;` n'est pas directement appliqué à `*` , vous pouvez donc facilement écraser cette propriété sur des éléments individuels.

Lire Le modèle de boîte en ligne: https://riptutorial.com/fr/css/topic/646/le-modele-de-boite

Chapitre 30: Les fonctions

Syntaxe

- `<calc()> = calc( <calc-sum> )`
- `<calc-sum> = <calc-product> [ [ '+' | '-' ] <calc-product> ]*`
- `<calc-product> = <calc-value> [ '*' <calc-value> | '/' <number> ]*`
- `<calc-value> = <number> | <dimension> | <percentage> | ( <calc-sum> )`

Remarques

Pour `calc()` , un espace blanc est requis autour des opérateurs " `-` " et " `+` ", mais pas les opérateurs " `*` " ou " `/` ".

Toutes les unités doivent être du même type. essayer de multiplier une hauteur par une durée, par exemple, est invalide.

Examples

fonction calc ()

Accepte une expression mathématique et renvoie une valeur numérique.

Cela est particulièrement utile lorsque vous travaillez avec différents types d'unités (par exemple, en soustrayant une valeur px d'un pourcentage) pour calculer la valeur d'un attribut.

`*` opérateurs `+` , `-` , `/` et `*` peuvent tous être utilisés, et des parenthèses peuvent être ajoutées pour spécifier l'ordre des opérations si nécessaire.

Utilisez `calc()` pour calculer la largeur d'un élément div:

```
#div1 {
    position: absolute;
    left: 50px;
    width: calc(100% - 100px);
    border: 1px solid black;
    background-color: yellow;
    padding: 5px;
    text-align: center;
}
```

Utilisez `calc()` pour déterminer la position d'une image d'arrière-plan:

```
background-position: calc(50% + 17px) calc(50% + 10px), 50% 50%;
```

Utilisez `calc()` pour déterminer la hauteur d'un élément:

```
height: calc(100% - 20px);
```

fonction attr ()

Renvoie la valeur d'un attribut de l'élément sélectionné.

Vous trouverez ci-dessous un élément blockquote qui contient un caractère à l'intérieur d'un attribut `data-*` que CSS peut utiliser (par exemple, à l'intérieur du pseudo-élément `::before` et `::after`) en utilisant cette fonction.

```
<blockquote data-mark='"'></blockquote>
```

Dans le bloc CSS suivant, le caractère est ajouté avant et après le texte à l'intérieur de l'élément:

```
blockquote[data-mark]::before,
blockquote[data-mark]::after {
    content: attr(data-mark);
}
```

fonction gradient linéaire ()

Crée une image représentant un dégradé linéaire de couleurs.

```
linear-gradient( 0deg, red, yellow 50%, blue);
```

Cela crée un dégradé allant du bas vers le haut, avec des couleurs commençant au rouge, puis au jaune à 50% et se terminant en bleu.

fonction de gradient radial ()

Crée une image représentant un dégradé de couleurs rayonnant depuis le centre du dégradé

```
radial-gradient(red, orange, yellow) /*A gradient coming out from the middle of the
gradient, red at the center, then orange, until it is finally yellow at the edges*/
```

fonction var ()

La fonction var () permet d'accéder aux variables CSS.

```
/* set a variable */
:root {
    --primary-color: blue;
}

/* access variable */
selector {
    color: var(--primary-color);
}
```

Cette fonctionnalité est en cours de développement. Consultez caniuse.com pour obtenir la dernière prise en charge du navigateur.

Lire Les fonctions en ligne: https://riptutorial.com/fr/css/topic/2214/les-fonctions

Chapitre 31: Les marges

Syntaxe

- marge: *<haut et droite et bas et gauche>* ;
- margin: *<top>* , *<left & right>* , *<bottom>* ;
- marge: *<haut et bas>* , *<gauche et droite>* ;
- margin: *<top>* , *<right>* , *<bottom>* , *<left>* ;
- margin-top: *<top>* ;
- margin-right: *<right>* ;
- margin-bottom: *<bas>* ;
- margin-left: *<gauche>* ;

Paramètres

Paramètre	Détails
0	définir la marge à aucun
auto	utilisé pour le centrage, en réglant uniformément les valeurs de chaque côté
unités (par exemple, px)	voir la section des paramètres dans Unités pour la liste des unités valides
hériter	hériter de la valeur de marge de l'élément parent
initiale	rétablir la valeur initiale

Remarques

Plus sur "Collapsing Margins": ici .

Examples

Appliquer la marge d'un côté donné

Propriétés spécifiques à la direction

CSS vous permet de spécifier un côté donné pour appliquer des marges. Les quatre propriétés fournies à cet effet sont les suivantes:

- `margin-left`

- `margin-right`
- `margin-top`
- `margin-bottom`

Le code suivant appliquerait une marge de 30 pixels sur le côté gauche du div sélectionné. *Voir résultat*

HTML

```
<div id="myDiv"></div>
```

CSS

```
#myDiv {
    margin-left: 30px;
    height: 40px;
    width: 40px;
    background-color: red;
}
```

Paramètre	Détails
marge à gauche	La direction dans laquelle la marge doit être appliquée.
30px	La largeur de la marge.

Spécification de la direction à l'aide d'une propriété raccourcie

La propriété de `margin` standard peut être développée pour spécifier différentes largeurs de chaque côté des éléments sélectionnés. La syntaxe pour ce faire est la suivante:

```
margin: <top> <right> <bottom> <left>;
```

L'exemple suivant applique une marge de largeur zéro au sommet du div, une marge de 10 pixels vers la droite, une marge de 50 pixels vers la gauche et une marge de 100 pixels vers la gauche. *Voir résultat*

HTML

```
<div id="myDiv"></div>
```

CSS

```
#myDiv {
    margin: 0 10px 50px 100px;
    height: 40px;
```

```
    width: 40px;
    background-color: red;
}
```

Effondrement de la marge

Lorsque deux marges se touchent verticalement, elles sont réduites. Lorsque deux marges se touchent horizontalement, elles ne se réduisent pas.

Exemple de marges verticales adjacentes:

Considérez les styles et les balises suivants:

```
div{
    margin: 10px;
}
```

```
<div>
    some content
</div>
<div>
    some more content
</div>
```

Ils seront séparés de 10px, car les marges verticales s'effondrent sur l'une et l'autre. (L'espacement ne sera pas la somme de deux marges.)

Exemple de marges horizontales adjacentes:

Considérez les styles et les balises suivants:

```
span{
    margin: 10px;
}
```

```
<span>some</span><span>content</span>
```

Ils seront séparés de 20px, car les marges horizontales ne s'effondrent pas sur l'une et l'autre. (L'espacement sera la somme de deux marges.)

Chevauchement avec différentes tailles

```
.top{
    margin: 10px;
}
.bottom{
    margin: 15px;
}
```

```
<div class="top">
    some content
```

```
</div>
<div class="bottom">
    some more content
</div>
```

Ces éléments seront espacés de 15px à la verticale. Les marges se chevauchent autant que possible, mais la marge la plus grande détermine l'espacement entre les éléments.

Marge de chevauchement

```
.outer-top{
    margin: 10px;
}
.inner-top{
    margin: 15px;
}
.outer-bottom{
    margin: 20px;
}
.inner-bottom{
    margin: 25px;
}
```

```
<div class="outer-top">
    <div class="inner-top">
        some content
    </div>
</div>
<div class="outer-bottom">
    <div class="inner-bottom">
        some more content
    </div>
</div>
```

Quel sera l'espacement entre les deux textes? (survolez pour voir la réponse)

> L'espacement sera de 25px. Comme les quatre marges se touchent, elles s’effondrent, utilisant ainsi la plus grande marge des quatre.

Maintenant, qu'en est-il si nous ajoutons des bordures au balisage ci-dessus.

```
div{
    border: 1px solid red;
}
```

Quel sera l'espacement entre les deux textes? (survolez pour voir la réponse)

> L'espacement sera de 59px! Désormais, seules les marges de .outer-top et .outer-bottom se touchent et sont les seules marges réduites. Les marges restantes sont séparées par les bordures. Nous avons donc 1px + 10px + 1px + ~~15px~~ + 20px + 1px + 25px + 1px. (Les 1px sont les frontières ...)

Réduire les marges entre les éléments parent et enfant:

HTML:

```
<h1>Title</h1>
<div>
  <p>Paragraph</p>
</div>
```

CSS

```
h1 {
  margin: 0;
  background: #cff;
}
div {
  margin: 50px 0 0 0;
  background: #cfc;
}
p {
  margin: 25px 0 0 0;
  background: #cf9;
}
```

Dans l'exemple ci-dessus, seule la plus grande marge s'applique. Vous vous êtes peut-être attendu à ce que le paragraphe soit situé à 60px du h1 (puisque l'élément div a une marge supérieure à 40px et que le p a une marge supérieure à 20px). Cela ne se produit pas parce que les marges s'effondrent pour former une seule marge.

Centrer horizontalement les éléments sur une page en utilisant la marge

Tant que l'élément est un **bloc** et qu'il a une **valeur de largeur explicitement définie** , les marges peuvent être utilisées pour centrer les éléments de bloc sur une page horizontalement.

Nous ajoutons une valeur de largeur inférieure à la largeur de la fenêtre et la propriété auto de la marge distribue alors l'espace restant à gauche et à droite:

```
#myDiv {
 width:80%;
 margin:0 auto;
}
```

Dans l'exemple ci - dessus , nous utilisons la sténographie `margin` déclaration d'abord fixer `0` aux valeurs de marge supérieure et inférieure (bien que cela puisse être une valeur) et nous utilisons `auto` pour laisser le navigateur allouer l'espace automatiquement aux valeurs de marge gauche et droite.

Dans l'exemple ci-dessus, l'élément #myDiv est défini sur 80% de largeur, ce qui laisse 20% de données restantes. Le navigateur distribue cette valeur aux côtés restants afin:

(100% - 80%) / 2 = 10%

Simplification de la marge

```
p {
    margin:1px;                 /* 1px margin in all directions */

    /*equals to:*/

    margin:1px 1px;

    /*equals to:*/

    margin:1px 1px 1px;

    /*equals to:*/

    margin:1px 1px 1px 1px;
}
```

Un autre exemple:

```
p{
    margin:10px 15px;         /* 10px margin-top & bottom And 15px margin-right & left*/

    /*equals to:*/

    margin:10px 15px 10px 15px;

    /*equals to:*/

    margin:10px 15px 10px;
    /* margin left will be calculated from the margin right value (=15px) */
}
```

Marges négatives

La marge est l'une des quelques propriétés CSS pouvant être définies sur des valeurs négatives. Cette propriété peut être utilisée pour **superposer des éléments sans positionnement absolu** .

```
div{
  display: inline;
}

#over{
  margin-left: -20px;
}

<div>Base div</div>
<div id="over">Overlapping div</div>
```

Exemple 1:

Il est évident de supposer que la valeur en pourcentage de la marge à la `margin-left` et à la `margin-right` serait relative à son élément parent.

```
.parent {
    width : 500px;
    height: 300px;
```

```
}

.child {
    width : 100px;
    height: 100px;
    margin-left: 10%;  /* (parentWidth * 10/100) => 50px */
}
```

Mais ce n'est pas le cas, en ce qui concerne la `margin-top` et la `margin-bottom` . Ces deux propriétés, en pourcentage, ne sont pas relatives à la hauteur du conteneur parent mais à la **largeur** du conteneur parent.

Alors,

```
.parent {
    width : 500px;
    height: 300px;
}

.child {
    width : 100px;
    height: 100px;
    margin-left: 10%;  /* (parentWidth * 10/100) => 50px  */
    margin-top: 20%;   /* (parentWidth * 20/100) => 100px */
}
```

Lire Les marges en ligne: https://riptutorial.com/fr/css/topic/305/les-marges

Chapitre 32: les tables

Syntaxe

- table-layout: *auto* | fixé;
- effondrement des frontières: *séparé* | effondrer;
- border-spacing: <longueur> | <longueur> <longueur>;
- cellules vides: *afficher* | cacher;
- côté légende: *haut* | bas;

Remarques

Ces propriétés s'appliquent aux deux éléments `<table>` (*) et HTML affichés sous forme d' `display: table` ou `display: inline-table`

(*) Les éléments `<table>` sont évidemment conçus de manière native par UA / navigers comme `display: table`

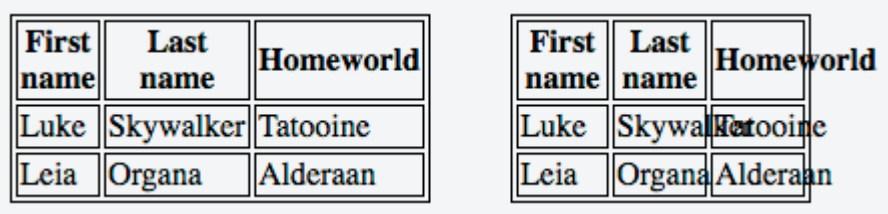
Les tables HTML sont sémantiquement valables pour les données tabulaires. Il n'est pas recommandé d'utiliser des tableaux pour la mise en page. Au lieu de cela, utilisez CSS.

Examples

mise en table

La propriété `table-layout` modifie l'algorithme utilisé pour la mise en page d'une table.

Ci-dessous un exemple de deux tables réglées sur la `width: 150px` :



La table de gauche a une disposition de `table-layout: auto` alors que celle de droite a une disposition de `table-layout: fixed` . Le premier est plus large que la largeur spécifiée (210 pixels au lieu de 150 pixels) mais le contenu convient. Ce dernier prend la largeur définie de 150px, que le contenu soit débordé ou non.

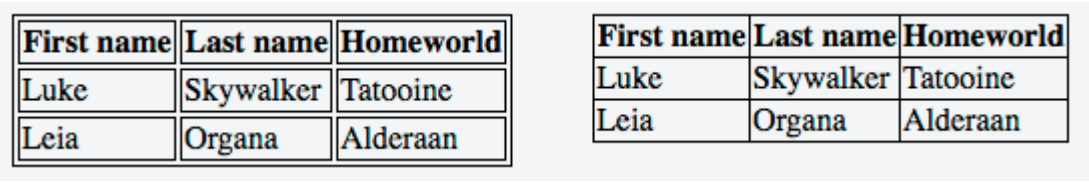
Valeur	La description
auto	Ceci est la valeur par défault. Il définit la disposition de la table à déterminer par le contenu de ses cellules.
fixé	Cette valeur définit la disposition de la table à déterminer par la propriété width

Valeur	La description
	fournie à la table. Si le contenu d'une cellule dépasse cette largeur, la cellule ne sera pas redimensionnée mais laissera le contenu déborder.

effondrement des frontières

La propriété `border-collapse` détermine si les bordures d'une table doivent être séparées ou fusionnées.

Ci-dessous un exemple de deux tables avec des valeurs différentes pour la propriété `border-collapse` :



First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

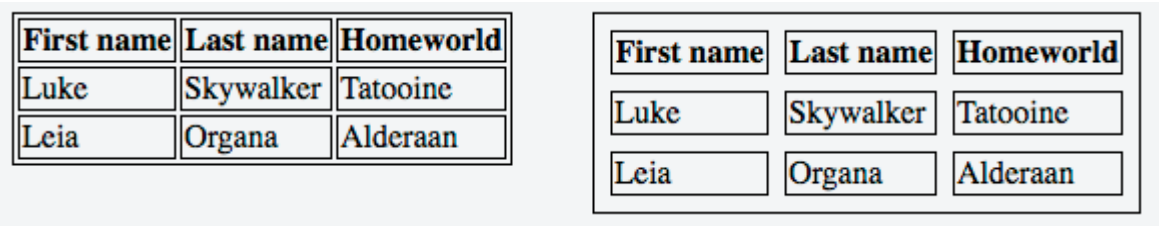
La table de gauche a une `border-collapse: separate` tandis que celle de droite a un `border-collapse: collapse` .

Valeur	La description
séparé	Ceci est la valeur par défault. Il sépare les bords de la table.
effondrer	Cette valeur définit les bordures de la table à fusionner, plutôt que d'être distincte.

espacement des frontières

La propriété d' `border-spacing` détermine l'espacement entre les cellules. Cela n'a aucun effet sauf si `border-collapse` est défini pour se `separate` .

Ci-dessous un exemple de deux tables avec des valeurs différentes pour la propriété d' `border-spacing` :



First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

La table de gauche a un `border-spacing: 2px` (valeur par défaut) tandis que celle de droite est `border-spacing: 8px` .

Valeur	La description
<longueur>	C'est le comportement par défaut, même si la valeur exacte peut varier d'un navigateur à l'autre.

Valeur	La description
<longueur> <longueur>	Cette syntaxe permet de spécifier des valeurs horizontales et verticales distinctes respectivement.

cellules vides

La propriété `empty-cells` détermine si les cellules sans contenu doivent être affichées ou non. Cela n'a aucun effet sauf si `border-collapse` est défini pour se `separate` .

Ci-dessous un exemple avec deux tables avec des valeurs différentes définies pour la propriété `empty-cells` :

First name	Last name	Homeworld
Luke		Tatooine
Leia	Organa	

First name	Last name	Homeworld
Luke		Tatooine
Leia	Organa	

La table de gauche a `empty-cells: show` alors que celle de droite a `empty-cells: hide` . Le premier affiche les cellules vides alors que le second ne les affiche pas.

Valeur	La description
montrer	Ceci est la valeur par défault. Il montre les cellules même si elles sont vides.
cacher	Cette valeur masque complètement une cellule s'il n'y a pas de contenu dans la cellule.

Plus d'information:

- https://www.w3.org/TR/CSS21/tables.html#empty-cells
- https://developer.mozilla.org/en-US/docs/Web/CSS/empty-cells
- http://codepen.io/SitePoint/pen/yfhtq
- https://css-tricks.com/almanac/properties/e/empty-cells/

côté légende

La propriété `caption-side` détermine le positionnement vertical de l'élément `<caption>` dans une table. Cela n'a aucun effet si un tel élément n'existe pas.

Sous un exemple avec deux tables avec des valeurs différentes définies pour la propriété `caption-side` :

Star Wars figures

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

First name	Last name	Homeworld
Luke	Skywalker	Tatooine
Leia	Organa	Alderaan

Star Wars figures

La table de gauche a une `caption-side: top` celle de droite a une `caption-side: bottom` .

Valeur	La description
Haut	Ceci est la valeur par défault. Il place la légende au-dessus de la table.
bas	Cette valeur place la légende sous la table.

Lire les tables en ligne: https://riptutorial.com/fr/css/topic/1074/les-tables

Chapitre 33: Modèle d'objet CSS (CSSOM)

Remarques

Le CSSOM (CSS Object Model) est une spécification à part entière.

Le projet actuel peut être trouvé ici: https://www.w3.org/TR/cssom-1/

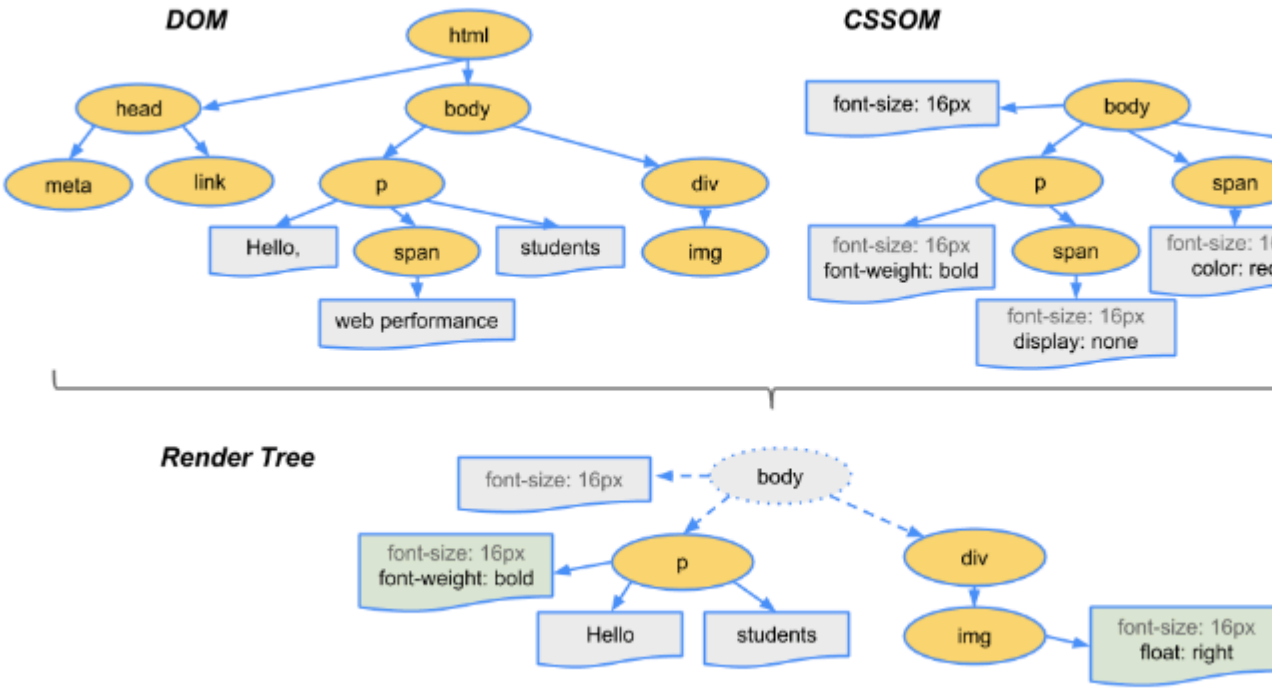
Examples

introduction

Le navigateur identifie les jetons de la feuille de style et les couvre en nœuds liés à une arborescence. La carte entière de tous les nœuds avec leurs styles associés d'une page serait le modèle d'objet CSS.

Pour afficher la page Web, un navigateur Web suit les étapes suivantes.

1. Le navigateur Web examine votre code HTML et construit le modèle DOM (Document Object Model).
2. Le navigateur Web examine votre CSS et construit le CSSOM (CSS Object Model).
3. Le navigateur Web combine le DOM et le CSSOM pour créer une arborescence de rendu. Le navigateur Web affiche votre page Web.



Ajouter une règle d'image d'arrière-plan via le CSSOM

Pour ajouter une règle d'image d'arrière-plan via le CSSOM, commencez par obtenir une référence aux règles de la première feuille de style:

```
var stylesheet = document.styleSheets[0].cssRules;
```

Ensuite, obtenez une référence à la fin de la feuille de style:

```
var end = stylesheet.length - 1;
```

Enfin, insérez une règle image d'arrière-plan pour l'élément body à la fin de la feuille de style:

```
stylesheet.insertRule("body { background-image:
url('http://cdn.sstatic.net/Sites/stackoverflow/img/favicon.ico'); }", end);
```

Lire Modèle d'objet CSS (CSSOM) en ligne: https://riptutorial.com/fr/css/topic/4961/modele-d-objet-css--cssom-

Chapitre 34: Modèles de conception CSS

Introduction

Ces exemples permettent de documenter des modèles de conception spécifiques à CSS tels que BEM , OOCSS et SMACSS .

Ces exemples ne sont PAS destinés à documenter des structures CSS telles que Bootstrap ou Foundation .

Remarques

Ces exemples servent à documenter des méthodologies / modèles de conception spécifiques à CSS.

Ces méthodologies incluent, mais ne sont pas exclusives aux suivantes:

- BEM
- OOCSS
- SMACSS

Ces exemples ne sont PAS destinés à documenter des structures CSS telles que Bootstrap ou Foundation . Bien que vous puissiez inclure des exemples d'application d'un ou de plusieurs modèles de méthodologie / conception CSS à un cadre CSS, ces exemples doivent se concentrer sur les méthodologies / modèles de conception avec ce cadre particulier et sur l'utilisation du cadre lui-même.

Examples

BEM

BEM signifie `Blocks, Elements and Modifiers` . Il s'agit d'une méthodologie conçue à l'origine par l'entreprise technologique russe Yandex , mais qui a également suscité un certain engouement chez les développeurs Web américains et d'Europe occidentale.

Comme le même implique, la méthololgie de BEM est tout au sujet de la composante de votre code HTML et CSS en trois types de composants:

- **Blocs:** entités autonomes significatives

  Les exemples sont en- `header` , `container` , `menu` , `checkbox` et `textbox`

- **Éléments:** Partie de blocs sans signification autonome et sémantiquement liés à leurs blocs.

  Exemples: `menu item list item` , `checkbox caption` et `header title`

- **Modificateurs:** Drapeaux sur un bloc ou un élément, utilisés pour modifier l'apparence ou le comportement

  Les exemples sont `disabled` , `highlighted` , `checked` , `fixed` , `size big` et `color yellow`

Le but de BEM est d'optimiser la lisibilité, la maintenabilité et la flexibilité de votre code CSS. Le moyen d'y parvenir est d'appliquer les règles suivantes.

- Les styles de bloc ne dépendent jamais d'autres éléments d'une page
- Les blocs doivent avoir un nom simple et court et éviter les caractères _ ou `-`
- Lorsque vous stylisez des éléments, utilisez des sélecteurs de format `blockname__elementname` format nom du `blockname__elementname`
- Lors de la mise en forme de modificateurs, utilisez des sélecteurs de format nom de `blockname--modifiername` et nom de `blockname__elementname--modifiername`
- Les éléments ou les blocs qui ont des modificateurs doivent hériter de tout élément du bloc ou de l'élément en cours de modification, à l'exception des propriétés que le modificateur est supposé modifier

Exemple de code

Si vous appliquez BEM à vos éléments de formulaire, vos sélecteurs CSS doivent ressembler à ceci:

```
.form { }                       // Block
.form--theme-xmas { }           // Block + modifier
.form--simple { }               // Block + modifier
.form__input { }                // Block > element
.form__submit { }               // Block > element
.form__submit--disabled { }     // Block > element + modifier
```

Le code HTML correspondant devrait ressembler à ceci:

```
<form class="form form--theme-xmas form--simple">
  <input class="form__input" type="text" />
  <input class="form__submit form__submit--disabled" type="submit" />
</form>
```

Lire Modèles de conception CSS en ligne: https://riptutorial.com/fr/css/topic/10823/modeles-de-conception-css

Chapitre 35: Normaliser les styles de navigateur

Introduction

Chaque navigateur dispose d'un ensemble par défaut de styles CSS qu'il utilise pour les éléments de rendu. Ces styles par défaut peuvent ne pas être cohérents entre les navigateurs car: les spécifications de langage ne sont pas claires, les styles de base sont donc à interpréter, les navigateurs peuvent ne pas respecter les spécifications données ou les styles par défaut Par conséquent, les utilisateurs peuvent vouloir normaliser les styles par défaut sur autant de navigateurs que possible.

Remarques

Bien que efficace, Meyer Reset apporte les mêmes modifications à presque tous les éléments couramment utilisés. Cela a pour effet de polluer les fenêtres d'inspecteur de navigateur Web avec les mêmes styles appliqués, et crée plus de travail pour le navigateur (plus de règles à appliquer à plusieurs éléments). La technique Normaliser, par contre, est beaucoup plus ciblée et moins technique. Cela simplifie le travail du navigateur et réduit l'encombrement des outils d'inspection du navigateur.

Examples

normalize.css

Les navigateurs utilisent un ensemble par défaut de styles CSS qu'ils utilisent pour les éléments de rendu. Certains de ces styles peuvent même être personnalisés en utilisant les paramètres du navigateur pour modifier les définitions de taille et de visage de police par défaut, par exemple. Les styles contiennent la définition des éléments supposés être au niveau du bloc ou en ligne, entre autres.

Étant donné que les styles de langage accordent une certaine latitude à ces styles par défaut et que les navigateurs ne respectent pas les spécifications, ils peuvent différer d’un navigateur à l’autre.

C'est là qu'intervient normalize.css . Il remplace les incohérences les plus courantes et corrige les bogues connus.

Qu'est ce que ça fait

- Préserve les valeurs par défaut utiles, contrairement à de nombreuses réinitialisations CSS.
- Normalise les styles pour un large éventail d'éléments.

- Corrige les bogues et les incohérences courantes du navigateur.
- Améliore la facilité d'utilisation avec des modifications subtiles.
- Explique ce que le code fait en utilisant des commentaires détaillés.

Ainsi, en incluant normalize.css dans votre projet, votre design se ressemblera et sera cohérent entre les différents navigateurs.

Différence à reset.css

Vous avez peut-être entendu parler de `reset.css` . Quelle est la différence entre les deux?

Alors que normalize.css fournit une cohérence en définissant différentes propriétés sur des valeurs par défaut unifiées, reset.css atteint la cohérence en **supprimant** tous les styles de base qu'un navigateur peut appliquer. Bien que cela puisse sembler une bonne idée au début, cela signifie en réalité que vous devez écrire **toutes les** règles vous-même, ce qui va à l'encontre de la norme.

Approches et exemples

Les réinitialisations CSS prennent des approches distinctes pour les paramètres par défaut du navigateur. Réinitialiser le CSS d'Eric Meyer existe depuis un certain temps. Son approche annule de nombreux éléments du navigateur connus pour causer des problèmes dès le départ. Ce qui suit provient de sa version (v2.0 | 20110126) Réinitialisation CSS.

```
html, body, div, span, applet, object, iframe,
h1, h2, h3, h4, h5, h6, p, blockquote, pre,
a, abbr, acronym, address, big, cite, code,
del, dfn, em, img, ins, kbd, q, s, samp,
small, strike, strong, sub, sup, tt, var,
b, u, i, center,
dl, dt, dd, ol, ul, li,
fieldset, form, label, legend,
table, caption, tbody, tfoot, thead, tr, th, td,
article, aside, canvas, details, embed,
figure, figcaption, footer, header, hgroup,
menu, nav, output, ruby, section, summary,
time, mark, audio, video {
    margin: 0;
    padding: 0;
    border: 0;
    font-size: 100%;
    font: inherit;
    vertical-align: baseline;
}
```

Réinitialiser le CSS d'Eric Meyer

Normaliser CSS de l'autre et traite plusieurs d'entre eux séparément. Ce qui suit est un exemple de la version (v4.2.0) du code.

```
/**
 * 1. Change the default font family in all browsers (opinionated).
```

```
 * 2. Correct the line height in all browsers.
 * 3. Prevent adjustments of font size after orientation changes in IE and iOS.
 */

/* Document
   ========================================================================== */

html {
  font-family: sans-serif; /* 1 */
  line-height: 1.15; /* 2 */
  -ms-text-size-adjust: 100%; /* 3 */
  -webkit-text-size-adjust: 100%; /* 3 */
}

/* Sections
   ========================================================================== */

/**
 * Remove the margin in all browsers (opinionated).
 */

body {
  margin: 0;
}

/**
 * Add the correct display in IE 9-.
 */

article,
aside,
footer,
header,
nav,
section {
  display: block;
}

/**
 * Correct the font size and margin on `h1` elements within `section` and
 * `article` contexts in Chrome, Firefox, and Safari.
 */

h1 {
  font-size: 2em;
  margin: 0.67em 0;
}
```

Normaliser CSS

Lire Normaliser les styles de navigateur en ligne:
https://riptutorial.com/fr/css/topic/1211/normaliser-les-styles-de-navigateur

Chapitre 36: Opacité

Syntaxe

- opacité: nombre (* strictement entre 0 et 1) | hériter | initiale | non défini;

Remarques

Si vous ne voulez pas appliquer d'opacité, vous pouvez l'utiliser à la place:

background: rgba (255, 255, 255, 0,6);

Ressources:

- MDN: https://developer.mozilla.org/en/docs/Web/CSS/opacity ;
- Transparence du W3C: la propriété 'opacité': https://www.w3.org/TR/css3-color/#transparency
- Support du navigateur: http://caniuse.com/#feat=css-opacity

Examples

Propriété d'opacité

L'opacité d'un élément peut être définie à l'aide de la propriété `opacity` . Les valeurs peuvent être comprises entre `0.0` (transparent) et `1.0` (opaque).

Exemple d'utilisation

```
<div style="opacity:0.8;">
    This is a partially transparent element
</div>
```

Valeur de la propriété	Transparence
`opacity: 1.0;`	Opaque
`opacity: 0.75;`	25% transparent (75% opaque)
`opacity: 0.5;`	50% transparent (50% opaque)
`opacity: 0.25;`	75% transparent (25% opaque)
`opacity: 0.0;`	Transparent

Compatibilité d'IE pour l'opacité

Pour utiliser l' `opacity` dans toutes les versions d'IE, l'ordre est le suivant:

```
.transparent-element {
  /* for IE 8 & 9 */
  -ms-filter:"progid:DXImageTransform.Microsoft.Alpha(Opacity=60)"; // IE8
  /* works in IE 8 & 9 too, but also 5, 6, 7 */
  filter: alpha(opacity=60); // IE 5-7
  /* Modern Browsers */
  opacity: 0.6;
}
```

Lire Opacité en ligne: https://riptutorial.com/fr/css/topic/2864/opacite

Chapitre 37: Performance

Examples

Utilisez transform et opacity pour éviter la mise en page du trigger

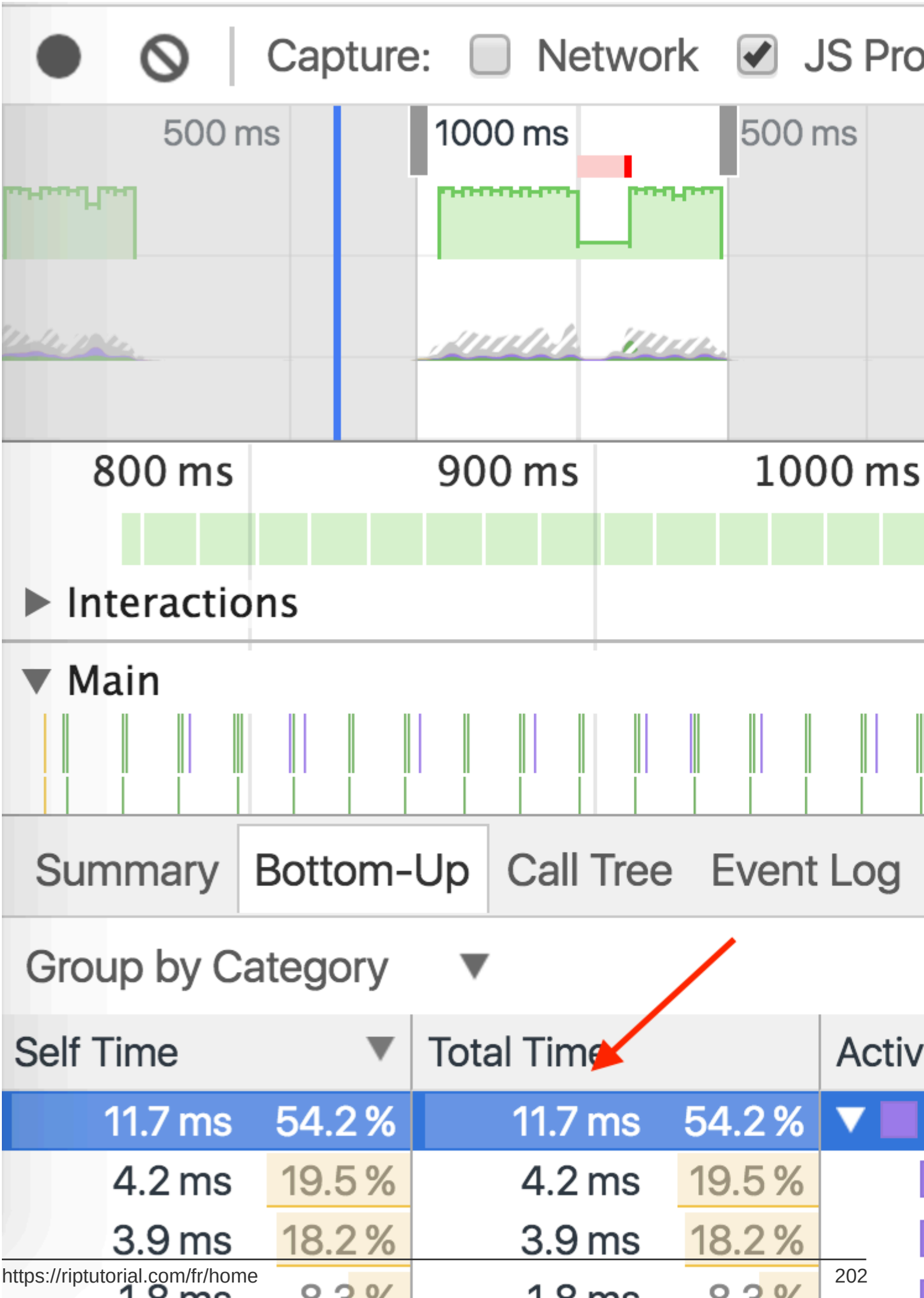
La modification de certains attributs CSS déclenchera le navigateur pour calculer de manière synchrone le style et la disposition, ce qui est une mauvaise chose lorsque vous devez animer à 60 images par seconde.

Ne pas

Animer avec la disposition des déclencheurs à `left` et en `top` .

```
#box {
  left: 0;
  top: 0;
  transition: left 0.5s, top 0.5s;
  position: absolute;
  width: 50px;
  height: 50px;
  background-color: gray;
}

#box.active {
  left: 100px;
  top: 100px;
}
```

La démo a pris **11,7 ms** pour le rendu, **9,8 ms** pour la peinture


Capture:
Network
JS Pro
500 ms
1000 ms
500 ms
800 ms
900 ms
1000 ms
Interactions
Main
Summary
Bottom-Up
Call Tree
Event Log
Group by Category
Self Time
Total Time
Activ
11.7 ms 54.2 %
11.7 ms 54.2 %
4.2 ms 19.5 %
4.2 ms 19.5 %
3.9 ms 18.2 %
3.9 ms 18.2 %
1.8 ms 8.3 %
1.8 ms 8.3 %

Chapitre 38: Plusieurs colonnes

Introduction

CSS permet de définir le contenu de l'élément en plusieurs colonnes avec des lacunes et des règles entre elles.

Remarques

Le niveau 1 du module de mise en forme multi-colonnes CSS est, depuis le 12 avril 2011, une recommandation du candidat W3C. Depuis lors, quelques petites modifications ont été apportées . Il est considéré comme étant dans l' étape Stable .

À compter du 3 juillet 2017, les navigateurs Internet Explorer 10 et 11 et Edge de Microsoft ne prennent en charge qu'une version antérieure de la spécification utilisant un préfixe de fournisseur.

Examples

Exemple de base

Considérez le balisage HTML suivant:

```
<section>
  <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum.</p>
  <p> Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet. Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.</p>
  <p>Lorem ipsum dolor sit amet, consetetur sadipscing elitr, sed diam nonumy eirmod tempor invidunt ut labore et dolore magna aliquyam erat, sed diam voluptua. At vero eos et accusam et justo duo dolores et ea rebum. Stet clita kasd gubergren, no sea takimata sanctus est Lorem ipsum dolor sit amet.</p>
</section>
```

Avec le CSS suivant, le contenu est divisé en trois colonnes séparées par une règle de colonne grise de deux pixels.

```
section {
  columns: 3;
  column-gap: 40px;
  column-rule: 2px solid gray;
}
```

Voir un échantillon en direct sur JSFiddle .

Créer plusieurs colonnes

```
<div class="content">
Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diam nonummy nibh
euismod tincidunt ut laoreet dolore magna aliquam erat volutpat. Ut wisi enim
ad minim veniam, quis nostrud exerci tation ullamcorper suscipit lobortis nisl
ut aliquip ex ea commodo consequat. Duis autem vel eum iriure dolor in
hendrerit in vulputate velit esse molestie consequat, vel illum dolore eu
feugiat nulla facilisis at vero eros et accumsan et iusto odio dignissim qui
blandit praesent luptatum zzril delenit augue duis dolore te feugait nulla
facilisi. Nam liber tempor cum soluta nobis eleifend option congue nihil
imperdiet doming id quod mazim placerat facer possim assum.
</div>
```

Css

```
.content {
-webkit-column-count: 3; /* Chrome, Safari, Opera */
-moz-column-count: 3; /* Firefox */
column-count: 3;
}
```

Lire Plusieurs colonnes en ligne: https://riptutorial.com/fr/css/topic/10688/plusieurs-colonnes

Chapitre 39: Positionnement

Syntaxe

- position: static | absolute | fixed | relative | sticky | initial | inherit | unset;
- z-index: auto | *nombre* | initial | hériter;

Paramètres

Paramètre	Détails
statique	Valeur par défaut. Les éléments s'affichent dans l'ordre, tels qu'ils apparaissent dans le flux de documents. Les propriétés top, right, bottom, left et z-index ne s'appliquent pas.
relatif	L'élément est positionné par rapport à sa position normale, donc à `left:20px` ajoute 20 pixels à la position GAUCHE de l'élément
fixé	L'élément est positionné par rapport à la fenêtre du navigateur
absolu	L'élément est positionné par rapport à son premier élément ancêtre positionné (pas statique)
initiale	Définit cette propriété sur sa valeur par défaut.
hériter	Hérite cette propriété de son élément parent.
gluant	Caractéristique expérimentale Il se comporte comme la `position: static` dans son parent jusqu'à ce qu'un seuil de décalage donné soit atteint, il agit alors comme une `position: fixed` .
non défini	Combinaison d'initiale et d'héritage. Plus d'infos ici .

Remarques

Le flux normal est le flux d'éléments si la position de l'élément est *statique* .

1. définir la *largeur* est bénéfique car dans certains cas, il empêche le chevauchement du contenu de l'élément.

Examples

Emploi stable

En définissant la position comme fixe, nous pouvons supprimer un élément du flux de documents et définir sa position par rapport à la fenêtre du navigateur. Une utilisation évidente est lorsque nous voulons que quelque chose soit visible lorsque nous faisons défiler vers le bas d'une longue page.

```
#stickyDiv {
    position:fixed;
    top:10px;
    left:10px;
}
```

Éléments superposés avec z-index

Pour modifier les éléments positionnés par ordre de pile par défaut (la propriété de `position` définie sur `relative` , `absolute` ou `fixed`), utilisez la propriété `z-index` .

Plus le z-index est élevé, plus il est placé dans le contexte d'empilement (sur l'axe des z).

Exemple

Dans l'exemple ci-dessous, un z-index de 3 met le vert en haut, un z-index de 2 met le rouge juste en dessous et un z-index de 1 met le bleu en dessous.

HTML

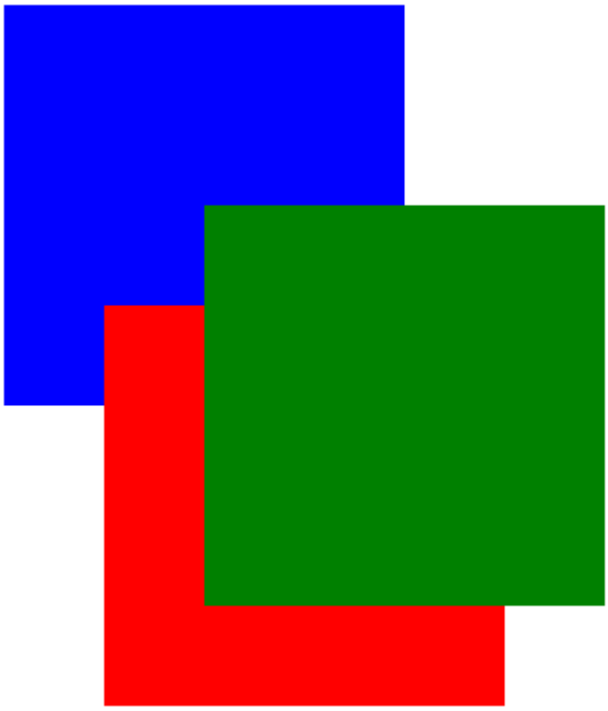
```
<div id="div1"></div>
<div id="div2"></div>
<div id="div3"></div>
```

CSS

```
div {
    position: absolute;
    height: 200px;
    width: 200px;
}
div#div1 {
    z-index: 1;
    left: 0px;
    top: 0px;
    background-color: blue;
}
div#div2 {
    z-index: 3;
    left: 100px;
    top: 100px;
    background-color: green;
}
div#div3 {
```

```
    z-index: 2;
    left: 50px;
    top: 150px;
    background-color: red;
}
```

Cela crée l'effet suivant:



Voir un exemple de travail à JSFiddle .

Syntaxe

```
z-index: [ number ] | auto;
```

Paramètre	Détails
`number`	Une valeur entière Un nombre plus élevé est plus élevé sur la pile d' `z-index` . `0` est la valeur par défaut. Les valeurs négatives sont autorisées.
`auto`	Donne à l'élément le même contexte d'empilement que son parent. (**Par défaut**)

Remarques

Tous les éléments sont disposés dans un axe 3D en CSS, y compris un axe de profondeur,

mesuré par la propriété `z-index` . `z-index` ne fonctionne que sur les éléments positionnés: (voir: Pourquoi z-index nécessite-t-il une position définie pour fonctionner?). La seule valeur ignorée est la valeur par défaut, `static` .

Lisez la propriété z-index et les contextes d'empilement dans la spécification CSS sur la présentation en couches et sur le réseau de développeurs Mozilla .

Position relative

Le positionnement relatif déplace l'élément par rapport à l'endroit où il se serait trouvé dans *les* propriétés *normales* .Offset:

1. Haut
2. la gauche
3. droite
4. bas

sont utilisés pour indiquer dans quelle mesure déplacer l'élément à partir duquel il se serait trouvé dans un flux normal.

```
.relpos{
    position:relative;
    top:20px;
    left:30px;
}
```

Ce code va déplacer la case contenant l'élément avec l'attribut class = "relpos" de 20px vers le bas et 30px vers la droite d'où il aurait été dans un flux normal.

Position absolue

Lorsque le positionnement absolu est utilisé, la case de l'élément souhaité est retirée du *flux normal* et n'affecte plus la position des autres éléments de la page. Propriétés de décalage:

1. Haut
2. la gauche
3. droite
4. bas

spécifie que l'élément doit apparaître par rapport à son élément contenant non statique suivant.

```
.abspos{
    position:absolute;
    top:0px;
    left:500px;
}
```

Ce code va déplacer la boîte contenant l'élément avec l'attribut `class="abspos"` vers le bas 0px et à droite 500px par rapport à son élément contenant.

Positionnement statique

La position par défaut d'un élément est `static` . Pour citer MDN :

> Ce mot-clé permet à l'élément d'utiliser le comportement normal, c'est-à-dire qu'il est disposé dans sa position actuelle dans le flux. Les propriétés top, right, bottom, left et z-index ne s'appliquent pas.

```
.element{
  position:static;
}
```

Lire Positionnement en ligne: https://riptutorial.com/fr/css/topic/935/positionnement

Chapitre 40: Propriété de filtre

Syntaxe

- filtre: aucun (valeur par défaut)
- filtre: initial (par défaut à aucun);
- filter: inherit (par défaut à la valeur parente);
- filtre: flou (px)
- filtre: luminosité (nombre |%)
- filtre: contraste (nombre |%)
- filtre: ombre portée (ombre horizontale-ombre-px-ombre-px-ombre-blur-px-propagation de la couleur)
- filtre: échelle de gris (nombre |%)
- filtre: teinte-rotation (deg)
- filtre: inverser (nombre |%)
- filtre: opacité (nombre |%)
- filtre: saturé (nombre |%)
- filtre: sépia (nombre |%)

Paramètres

Valeur	La description
flou (x)	Floue l'image par x pixels.
luminosité (x)	Éclaircit l'image à n'importe quelle valeur supérieure à 1,0 ou 100%. En dessous, l'image sera assombrie.
contraste (x)	Fournit plus de contraste à l'image pour toute valeur supérieure à 1,0 ou 100%. En dessous, l'image sera moins saturée.
ombre portée (h, v, x, y, z)	Donne à l'ombre une ombre portée. h et v peuvent avoir des valeurs négatives. x, y et z sont facultatifs.
échelle de gris (x)	Affiche l'image en niveaux de gris, avec une valeur maximale de 1,0 ou 100%.
teinte rotation (x)	Applique une teinte-rotation à l'image.
inverser (x)	Inverse la couleur de l'image avec une valeur maximale de 1,0 ou 100%.
opacité (x)	Définit l'opacité / la transparence de l'image avec une valeur maximale de 1,0 ou 100%.
saturer (x)	Sature l'image à n'importe quelle valeur supérieure à 1,0 ou 100%. En

Valeur	La description
	dessous, l'image commence à se désaturer.
sépia (x)	Convertit l'image en sépia avec une valeur maximale de 1,0 ou 100%.

Remarques

1. Puisque filter est une fonctionnalité expérimentale, vous devez utiliser le préfixe -webkit. Cela peut changer dans la syntaxe et le comportement, mais les changements vont probablement être petits.
2. Il peut ne pas être pris en charge dans les anciennes versions des principaux navigateurs. Il peut être entièrement non pris en charge dans les navigateurs mobiles.
3. En raison de son support relativement limité, essayez d'utiliser `box-shadow` au lieu de `filter: drop-shadow()` . Utilisez l' `opacity` au lieu du `filter: opacity()` .
4. Il peut être animé via Javascript / jQuery. Pour Javascript, utilisez `object.style.WebkitFilter` .
5. Vérifiez W3Schools ou MDN pour plus d'informations.
6. W3Schools a également une page de démonstration pour tous les différents types de valeurs de filtre.

Examples

Ombre portée (utilisez plutôt l'ombre de la boîte si possible)

HTML

```
<p>My shadow always follows me.</p>
```

CSS

```
p {
    -webkit-filter: drop-shadow(10px 10px 1px green);
    filter: drop-shadow(10px 10px 1px green);
}
```

Résultat

Plusieurs valeurs de filtre

Pour utiliser plusieurs filtres, séparez chaque valeur par un espace.

HTML

```
<img src='donald-duck.png' alt='Donald Duck' title='Donald Duck' />
```

CSS

```
img {
    -webkit-filter: brightness(200%) grayscale(100%) sepia(100%) invert(100%);
    filter: brightness(200%) grayscale(100%) sepia(100%) invert(100%);
}
```

Résultat

Teinte Rotation

HTML

```
<img src='donald-duck.png' alt='Donald Duck' title='Donald Duck' />
```

CSS

```
img {
    -webkit-filter: hue-rotate(120deg);
    filter: hue-rotate(120deg);
}
```

Résultat

Inverser la couleur

HTML

```
<div></div>
```

CSS

```
div {
    width: 100px;
    height: 100px;
    background-color: white;
    -webkit-filter: invert(100%);
    filter: invert(100%);
}
```

Résultat

Tourne du blanc au noir.

Brouiller

HTML

```
<img src='donald-duck.png' alt='Donald Duck' title='Donald Duck' />
```

CSS

```
img {
    -webkit-filter: blur(1px);
    filter: blur(1px);
}
```

Résultat

Tu veux frotter tes lunettes.

Lire Propriété de filtre en ligne: https://riptutorial.com/fr/css/topic/1567/propriete-de-filtre

Chapitre 41: Propriétés personnalisées (variables)

Introduction

Les variables CSS permettent aux auteurs de créer des valeurs réutilisables pouvant être utilisées dans un document CSS.

Par exemple, il est courant en CSS de réutiliser une seule couleur dans un document. Avant les variables CSS, cela impliquerait de réutiliser la même valeur de couleur plusieurs fois dans un document. Avec les variables CSS, la valeur de couleur peut être affectée à une variable et référencée à plusieurs endroits. Cela facilite la modification des valeurs et est plus sémantique que l'utilisation des valeurs CSS traditionnelles.

Syntaxe

- : pseudo-classe root {} / * permettant une définition plus globale des variables * /
- - nom-variable: *valeur* ; / * définir la variable * /
- var (- nom-variable, *valeur par défaut*) / * utilise une variable définie avec une valeur par défaut repli * /

Remarques

Les variables CSS sont actuellement considérées comme une technologie expérimentale.

SUPPORT DE NAVIGATEUR / COMPATIBILITÉ

Firefox: Version **31** + *(activé par défaut)*

Plus d'info de Mozilla

Chrome: Version **49** + *(activé par défaut)* .

"Cette fonctionnalité peut être activée dans Chrome Version 48 pour les tests en activant la fonctionnalité `experimental Web Platform` *. Entrez* `chrome://flags/` *dans votre barre d'adresse Chrome pour accéder à ce paramètre."*

IE: Non pris en charge.

Edge: en cours de développement

Safari: Version **9.1+**

Examples

Couleur variable

```
:root {
  --red: #b00;
  --blue: #4679bd;
  --grey: #ddd;
}
.Bx1 {
  color: var(--red);
  background: var(--grey);
  border: 1px solid var(--red);
}
```

Dimensions variables

```
:root {
  --W200: 200px;
  --W10: 10px;
}
.Bx2 {
  width: var(--W200);
  height: var(--W200);
  margin: var(--W10);
}
```

Variable en cascade

Les variables CSS se répercutent de la même manière que les autres propriétés et peuvent être retraitées en toute sécurité.

Vous pouvez définir des variables plusieurs fois et seule la définition avec la spécificité la plus élevée s'appliquera à l'élément sélectionné.

En supposant ce HTML:

```
<a class="button">Button Green</a>
<a class="button button_red">Button Red</a>
<a class="button">Button Hovered On</a>
```

Nous pouvons écrire ce CSS:

```
.button {
  --color: green;
  padding: .5rem;
  border: 1px solid var(--color);
  color: var(--color);
}

.button:hover {
  --color: blue;
```

```
}

.button_red {
  --color: red;
}
```

Et obtenez ce résultat:

Valide / Invalides

Nommer Lorsque vous nommez des variables CSS, il ne contient que des lettres et des tirets, comme les autres propriétés CSS (ex: line-height, -moz-box-sizing), mais il doit commencer par des doubles tirets (-).

```
//These are Invalids variable names
--123color: blue;
--#color: red;
--bg_color: yellow
--$width: 100px;

//Valid variable names
--color: red;
--bg-color: yellow
--width: 100px;
```

Les variables CSS sont sensibles à la casse.

```
/* The variable names below are all different variables */
--pcolor: ;
--Pcolor: ;
--pColor: ;
```

Vs Space vide

```
/* Invalid */
    --color:;

/* Valid */
  --color: ; /* space is assigned */
```

Concaténations

```
    /* Invalid - CSS doesn't support concatenation*/
    .logo{
        --logo-url: 'logo';
        background: url('assets/img/' var(--logo-url) '.png');
    }

    /* Invalid - CSS bug */
```

```
    .logo{
        --logo-url: 'assets/img/logo.png';
        background: url(var(--logo-url));
    }

    /* Valid */
    .logo{
        --logo-url: url('assets/img/logo.png');
        background: var(--logo-url);
    }
```

Attention lorsque vous utilisez des unités

```
    /* Invalid */
    --width: 10;
    width: var(--width)px;

    /* Valid */
    --width: 10px;
    width: var(--width);

    /* Valid */
    --width: 10;
    width: calc(1px * var(--width)); /* multiply by 1 unit to convert */
    width: calc(1em * var(--width));
```

Avec des requêtes médiatiques

Vous pouvez redéfinir les variables dans les requêtes multimédias et faire en sorte qu'elles soient utilisées en cascade, ce qui n'est pas possible avec les variables de préprocesseur.

Ici, une requête média modifie les variables utilisées pour configurer une grille très simple:

HTML

```
<div></div>
<div></div>
<div></div>
<div></div>
```

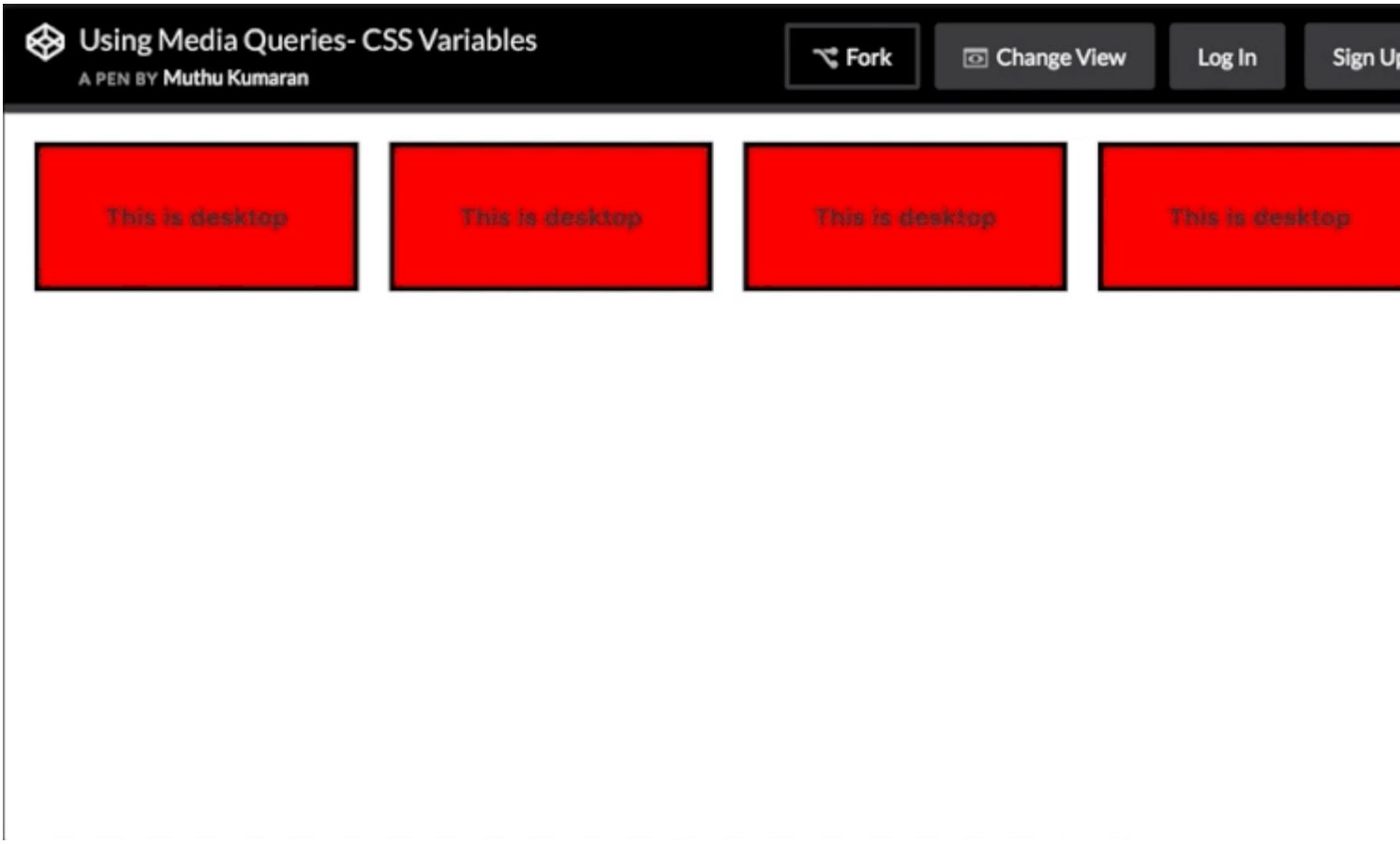
CSS

```
:root{
    --width: 25%;
    --content: 'This is desktop';
}
@media only screen and (max-width: 767px){
    :root{
        --width:50%;
        --content: 'This is mobile';
    }
}
@media only screen and (max-width: 480px){
    :root{
        --width:100%;
```

```
    }
}

div{
    width: calc(var(--width) - 20px);
    height: 100px;
}
div:before{
    content: var(--content);
}

/* Other Styles */
body {
    padding: 10px;
}

div{
    display: flex;
    align-items: center;
    justify-content: center;
    font-weight:bold;
    float:left;
    margin: 10px;
    border: 4px solid black;
    background: red;
}
```

Vous pouvez essayer de redimensionner la fenêtre dans cette démonstration CodePen

Voici une capture d'écran animée du redimensionnement en action:



Lire Propriétés personnalisées (variables) en ligne:
https://riptutorial.com/fr/css/topic/1755/proprietes-personnalisees--variables-

Chapitre 42: Pseudo-éléments

Introduction

Les pseudo-éléments, tout comme les pseudo-classes, sont ajoutés à un sélecteur CSS, mais au lieu de décrire un état spécial, ils vous permettent d'étendre et de styliser certaines parties d'un élément HTML.

Par exemple, le pseudo-élément :: first-letter ne cible que la première lettre d'un élément de bloc spécifié par le sélecteur.

Syntaxe

- selector :: pseudo-element {property: value}

Paramètres

pseudo-élément	La description
`::after`	Insérer du contenu après le contenu d'un élément
`::before`	Insérer du contenu avant le contenu d'un élément
`::first-letter`	Sélectionne la première lettre de chaque élément
`::first-line`	Sélectionne la première ligne de chaque élément
`::selection`	Correspond à la partie d'un élément sélectionnée par un utilisateur
`::backdrop`	Utilisé pour créer une toile de fond qui masque le document sous-jacent pour un élément de la pile de la couche supérieure
`::placeholder`	Vous permet de styliser le texte d'espace réservé d'un élément de formulaire (expérimental)
`::marker`	Pour appliquer des attributs de style liste sur un élément donné (expérimental)
`::spelling-error`	Représente un segment de texte signalé incorrectement par le navigateur (Expérimental)
`::grammar-error`	Représente un segment de texte que le navigateur a identifié comme grammaticalement incorrect (Expérimental)

Remarques

- Parfois , vous verrez doubles points (`::`) au lieu d'un seul (`:`). Ceci est un moyen de séparer les pseudo-classes de pseudo-éléments, mais certains anciens navigateurs comme Internet Explorer 8 **ne** supporte **que** deux points simple (`:`) pour les pseudo-éléments.
- On ne peut utiliser qu'un seul pseudo-élément dans un sélecteur. Il doit apparaître après les sélecteurs simples dans la déclaration.
- Les pseudo-éléments ne font pas partie du DOM et ne peuvent donc pas être ciblés par `:hover` ou d'autres événements utilisateur.

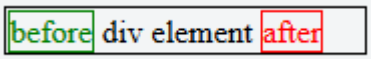
Examples

Pseudo-éléments

Les pseudo-éléments sont ajoutés aux sélecteurs mais, au lieu de décrire un état spécial, ils vous permettent de styliser certaines parties d'un document.

L'attribut `content` est requis pour les pseudo-éléments à rendre; Cependant, l'attribut peut avoir une valeur vide (par exemple, `content: ""`).

```
div::after {
  content: 'after';
  color: red;
  border: 1px solid red;
}

div {
  color: black;
  border: 1px solid black;
  padding: 1px;
}

div::before {
  content: 'before';
  color: green;
  border: 1px solid green;
}
```



Pseudo-éléments dans les listes

Les pseudo-éléments sont souvent utilisés pour modifier l'apparence des listes (principalement pour les listes non ordonnées, `ul`).

La première étape consiste à supprimer les puces de liste par défaut:

```
ul {
```

```
  list-style-type: none;
}
```

Ensuite, vous ajoutez le style personnalisé. Dans cet exemple, nous allons créer des boîtes de dégradés pour les puces.

```
li:before {
  content: "";
  display: inline-block;
  margin-right: 10px;
  height: 10px;
  width: 10px;
  background: linear-gradient(red, blue);
}
```

HTML

```
<ul>
   <li>Test I</li>
   <li>Test II</li>
</ul>
```

Résultat

Lire Pseudo-éléments en ligne: https://riptutorial.com/fr/css/topic/911/pseudo-elements

Chapitre 43: Rembourrage

Syntaxe

- padding: *length* | initial | inherit | unset;
- padding-top: *longueur* | initial | inherit | unset;
- padding-right: *length* | initial | inherit | unset;
- padding-bottom: *length* | initial | inherit | unset;
- padding-left: *length* | initial | inherit | unset;

Remarques

> La propriété padding définit l'espace de remplissage de tous les côtés d'un élément. La zone de remplissage est l'espace entre le contenu de l'élément et sa bordure. **Les valeurs négatives ne sont pas autorisées** .

1 : https://developer.mozilla.org/en/docs/Web/CSS/padding MDN

Voir aussi cette question , "Pourquoi CSS ne supporte-t-il pas le remplissage négatif?" et ses réponses.

Veuillez également considérer le modèle de boîte lors de l'utilisation du remplissage. Selon la valeur de la taille de la boîte, le remplissage d'un élément peut soit ajouter la hauteur / largeur précédemment définie d'un élément ou non.

Propriétés connexes:

marge

Le remplissage sur des éléments en ligne ne s'appliquera qu'à gauche et à droite de l'élément, et non au haut et au bas, en raison des propriétés d'affichage inhérentes aux éléments en ligne.

Examples

Rembourrage d'un côté donné

La propriété padding définit l'espace de remplissage de tous les côtés d'un élément. La zone de remplissage est l'espace entre le contenu de l'élément et sa bordure. Les valeurs négatives ne sont pas autorisées.

Vous pouvez spécifier un côté individuellement:

- `padding-top`
- `padding-right`
- `padding-bottom`
- `padding-left`

Le code suivant ajouterait un remplissage de `5px` en haut de la div:

```
<style>
.myClass {
    padding-top: 5px;
}
</style>

<div class="myClass"></div>
```

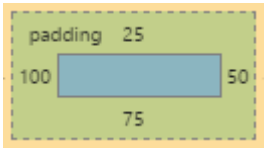
Rembourrage

La propriété padding définit l'espace de remplissage de tous les côtés d'un élément. La zone de remplissage est l'espace entre le contenu de l'élément et sa bordure. Les valeurs négatives ne sont pas autorisées.

Pour éviter d'ajouter du rembourrage à chaque côté individuellement (en utilisant un `padding-top` , un `padding-left` etc.), pouvez-vous l'écrire comme ci-dessous:

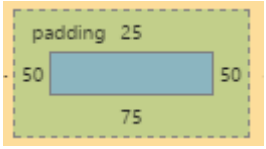
Quatre valeurs :

```
<style>
    .myDiv {
        padding: 25px 50px 75px 100px; /* top right bottom left; */
    }
</style>
<div class="myDiv"></div>
```



Trois valeurs :

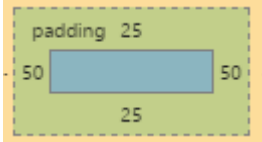
```
<style>
    .myDiv {
        padding: 25px 50px 75px; /* top left/right bottom */
    }
</style>
<div class="myDiv"></div>
```



Deux valeurs :

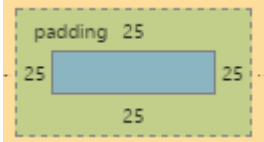
```
<style>
    .myDiv {
        padding: 25px 50px; /* top/bottom left/right */
    }
```

```
</style>
<div class="myDiv"></div>
```



Une valeur :

```
<style>
    .myDiv {
        padding: 25px; /* top/right/bottom/left */
    }
</style>
<div class="myDiv"></div>
```



Lire Rembourrage en ligne: https://riptutorial.com/fr/css/topic/1255/rembourrage

Chapitre 44: Requêtes médias

Syntaxe

- @media [pas | seulement] mediatype et (fonction média) {/ * Règles CSS à appliquer * /}

Paramètres

Paramètre	Détails
`mediatype`	(Facultatif) C'est le type de support. Pourrait être n'importe quoi dans la gamme de `all` à l' `screen` .
`not`	(Facultatif) N'applique pas le CSS pour ce type de média particulier et s'applique à tout le reste.
`media feature`	Logique pour identifier les cas d'utilisation pour CSS. Options décrites ci-dessous.
Fonction média	**Détails**
`aspect-ratio`	Décrit le rapport d'aspect de la zone d'affichage ciblée du périphérique de sortie.
`color`	Indique le nombre de bits par composant couleur du périphérique de sortie. Si le périphérique n'est pas un périphérique couleur, cette valeur est égale à zéro.
`color-index`	Indique le nombre d'entrées dans la table de couleurs pour le périphérique de sortie.
`grid`	Détermine si le périphérique de sortie est un périphérique de grille ou un périphérique bitmap.
`height`	La fonction de support de hauteur décrit la hauteur de la surface de rendu du périphérique de sortie.
`max-width`	CSS ne s'appliquera pas sur une largeur d'écran supérieure à celle spécifiée.
`min-width`	CSS ne s'appliquera pas sur une largeur d'écran inférieure à celle spécifiée.
`max-height`	CSS ne s'appliquera pas à une hauteur d'écran plus grande que celle spécifiée.
`min-height`	CSS ne s'appliquera pas sur une hauteur d'écran plus courte que celle

Paramètre	Détails
	spécifiée.
`monochrome`	Indique le nombre de bits par pixel sur un périphérique monochrome (échelle de gris).
`orientation`	CSS affichera uniquement si le périphérique utilise l'orientation spécifiée. Voir les remarques pour plus de détails.
`resolution`	Indique la résolution (densité de pixels) du périphérique de sortie.
`scan`	Décrit le processus de numérisation des périphériques de sortie de télévision.
`width`	La fonctionnalité de support de largeur décrit la largeur de la surface de rendu du périphérique de sortie (telle que la largeur de la fenêtre du document ou la largeur du bloc de page sur une imprimante).
Fonctionnalités obsolètes	**Détails**
`device-aspect-ratio`	**`Deprecated`** CSS **`Deprecated`** ne s'affichent que sur les appareils dont le rapport hauteur / largeur correspond au ratio spécifié. Ceci est une fonctionnalité `deprecated` et il n'est pas garanti que cela fonctionne.
`max-device-width`	**`Deprecated`** Même que `max-width` mais mesure la largeur de l'écran physique, plutôt que la largeur d'affichage du navigateur.
`min-device-width`	**`Deprecated`** Même que `min-width` mais mesure la largeur de l'écran physique plutôt que la largeur d'affichage du navigateur.
`max-device-height`	**`Deprecated`** Même que `max-height` mais mesure la largeur de l'écran physique plutôt que la largeur d'affichage du navigateur.
`min-device-height`	**`Deprecated`** Même que `min-height` mais mesure la largeur de l'écran physique, plutôt que la largeur d'affichage du navigateur.

Remarques

Les requêtes multimédias sont prises en charge dans tous les navigateurs modernes, y compris Chrome, Firefox, Opera et Internet Explorer 9 et versions ultérieures.

Il est important de noter que la fonction de support d' `orientation` n'est pas limitée aux périphériques mobiles. Il est basé sur la largeur et la hauteur de la fenêtre d'affichage (pas la fenêtre ou les périphériques).

Le mode Paysage est utilisé lorsque la largeur de la fenêtre d'affichage est supérieure à la

hauteur de la fenêtre d'affichage.

Le mode Portrait est utilisé lorsque la hauteur de la fenêtre d'affichage est supérieure à la largeur de la fenêtre d'affichage.

Cela se traduit généralement par un moniteur de bureau en mode paysage, mais peut-il parfois être portrait.

Dans la plupart des cas, les appareils mobiles signaleront leur résolution et non leur taille réelle en pixels, qui peut différer en raison de la densité des pixels. Pour les forcer à déclarer leur taille réelle de pixels ajouter les éléments suivants dans votre `head` tag:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Examples

Exemple de base

```
@media screen and (min-width: 720px) {
    body {
        background-color: skyblue;
    }
}
```

La requête multimédia ci-dessus spécifie deux conditions:

1. La page doit être visualisée sur un écran normal (pas une page imprimée, un projecteur, etc.).
2. La largeur du port d'affichage de l'utilisateur doit être d'au moins 720 pixels.

Si ces conditions sont remplies, les styles à l'intérieur de la requête multimédia seront actifs et la couleur d'arrière-plan de la page sera bleu ciel.

Les requêtes multimédias sont appliquées dynamiquement. Si, lors du chargement de la page, les conditions spécifiées dans la requête de média sont remplies, le CSS sera appliqué, mais sera immédiatement désactivé si les conditions ne sont plus remplies. À l'inverse, si les conditions ne sont pas remplies initialement, le CSS ne sera pas appliqué tant que les conditions spécifiées ne sont pas remplies.

Dans notre exemple, si la largeur du port d'affichage de l'utilisateur est initialement supérieure à 720 pixels, mais que l'utilisateur réduit la largeur du navigateur, la couleur d'arrière-plan cessera d'être bleu ciel dès que l'utilisateur aura redimensionné le port d'affichage à moins de 720 pixels. largeur.

Utiliser sur la balise link

```
<link rel="stylesheet" media="min-width: 600px" href="example.css" />
```

Cette feuille de style est toujours téléchargée, mais elle est appliquée uniquement aux périphériques dont la largeur d'écran est supérieure à 600 pixels.

type de support

Les requêtes de média ont un paramètre optionnel `mediatype` . Ce paramètre est placé directement après la `@media` déclaration (`@media mediatype`), par exemple:

```
@media print {
    html {
        background-color: white;
    }
}
```

Le code CSS ci-dessus donnera à l'élément `HTML` DOM une couleur de fond blanche lors de l'impression.

Le paramètre `mediatype` a un préfixe facultatif `not` ou `only` qui appliquera les styles à tout sauf le type de média spécifié *ou* uniquement le type de média spécifié. Par exemple, l'exemple de code suivant appliquera le style à chaque type de support, à l'exception de l' `print` .

```
@media not print {
    html {
        background-color: green;
    }
}
```

Et de la même manière, pour le montrer uniquement à l'écran, cela peut être utilisé:

```
@media only screen {
    .fadeInEffects {
        display: block;
    }
}
```

La liste de `mediatype` peut être mieux comprise avec le tableau suivant:

Type de support	La description
`all`	Appliquer à tous les appareils
`screen`	Ordinateurs par défaut
`print`	Imprimantes en général. Utilisé pour mettre en forme les versions imprimées des sites Web
`handheld`	PDA, téléphones portables et appareils portables avec un petit écran
`projection`	Pour une présentation projetée, par exemple des projecteurs

Type de support	La description
`aural`	Systèmes de parole
`braille`	Appareils tactiles en braille
`embossed`	Imprimantes braille paginées
`tv`	Appareils de type télévision
`tty`	Dispositifs avec une grille de caractères à pas fixe. Terminaux, portables.

Utilisation de requêtes multimédia pour cibler différentes tailles d'écran

Souvent, la conception Web réactive implique des requêtes média, qui sont des blocs CSS exécutés uniquement si une condition est remplie. Ceci est utile pour la conception Web réactive car vous pouvez utiliser des requêtes multimédia pour spécifier différents styles CSS pour la version mobile de votre site Web par rapport à la version de bureau.

```
@media only screen and (min-width: 300px) and (max-width: 767px) {
    .site-title {
        font-size: 80%;
    }

    /* Styles in this block are only applied if the screen size is atleast 300px wide, but no
more than 767px */
}

@media only screen and (min-width: 768px) and (max-width: 1023px) {
    .site-title {
        font-size: 90%;
    }

    /* Styles in this block are only applied if the screen size is atleast 768px wide, but no
more than 1023px */
}

@media only screen and (min-width: 1024px) {
    .site-title {
        font-size: 120%;
    }

    /* Styles in this block are only applied if the screen size is over 1024px wide. */
}
```

Largeur vs Viewport

Lorsque nous utilisons "width" avec des requêtes multimédias, il est important de définir correctement la balise meta. La balise META de base ressemble à ceci et doit être placée dans la `<head>` .

```
<meta name="viewport" content="width=device-width,initial-scale=1">
```

Pourquoi c'est important?

Basé sur la définition de MDN, "width" est

> La fonctionnalité de support de largeur décrit la largeur de la surface de rendu du périphérique de sortie (telle que la largeur de la fenêtre du document ou la largeur du bloc de page sur une imprimante).

Qu'est-ce que ça veut dire?

View-port est la largeur de l'appareil lui-même. Si votre résolution d'écran indique que la résolution est de 1280 x 720, la largeur de votre port d'affichage est "1280px".

Plus souvent, de nombreux appareils allouent une quantité de pixels différente pour afficher un pixel. Par exemple, l'iPhone 6 Plus a une résolution de 1242 x 2208. Mais la largeur et la hauteur de la fenêtre d'affichage sont de 414 x 736. Cela signifie que 3 pixels sont utilisés pour créer 1 pixel.

Mais si vous n'avez pas défini la balise `meta` correctement, elle essaiera d'afficher votre page Web avec sa résolution native, ce qui se traduira par un zoom arrière (textes et images plus petits).

Requêtes multimédias pour les écrans de rétine et non rétine

Bien que cela ne fonctionne que pour les navigateurs WebKit, ceci est utile:

```
/* ----------- Non-Retina Screens ----------- */
@media screen
  and (min-width: 1200px)
  and (max-width: 1600px)
  and (-webkit-min-device-pixel-ratio: 1) {
}

/* ----------- Retina Screens ----------- */
@media screen
  and (min-width: 1200px)
  and (max-width: 1600px)
  and (-webkit-min-device-pixel-ratio: 2)
  and (min-resolution: 192dpi) {
}
```

Informations d'arrière-plan

Il y a deux types de pixels dans l'affichage. L'un est les pixels logiques et l'autre les pixels physiques. La plupart du temps, les pixels physiques restent les mêmes, car il en va de même pour tous les périphériques d'affichage. Les pixels logiques changent en fonction de la résolution des périphériques pour afficher des pixels de meilleure qualité. Le ratio de pixels de l'appareil est le rapport entre les pixels physiques et les pixels logiques. Par exemple, le MacBook Pro Retina, l'iPhone 4 et les versions ultérieures indiquent un rapport de pixels de périphérique de 2, car la résolution linéaire physique est le double de la résolution logique.

La raison pour laquelle cela fonctionne uniquement avec les navigateurs WebKit est due à:

- Le préfixe du fournisseur `-webkit-` avant la règle.
- Cela n'a pas été implémenté dans les moteurs autres que WebKit et Blink.

Terminologie et Structure

Requêtes médias permettent d'appliquer des règles CSS en fonction du type de périphérique / support (p.ex. écran, impression ou portable) appelé **type de média,** d' autres aspects du dispositif sont décrits avec des **fonctionnalités de médias** tels que la disponibilité des dimensions de couleur ou de fenêtre.

Structure générale d'une requête média

```
@media [...] {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

Une requête multimédia contenant un type de média

```
@media print {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

Une requête de média contenant un type de média et une fonctionnalité de média

```
@media screen and (max-width: 600px) {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

Une requête de média contenant une fonctionnalité de média (et un type de média implicite de "tous")

```
@media (orientation: portrait) {
    /* One or more CSS rules to apply when the query is satisfied */
}
```

Requêtes médiatiques et IE8

Les requêtes multimédias ne sont pas prises en charge dans IE8 et les versions ultérieures.

Une solution de contournement basée sur Javascript

Pour ajouter un support pour IE8, vous pouvez utiliser l'une des solutions JS. Par exemple, **Respond** peut être ajouté pour ajouter le support de requête de média pour IE8 uniquement avec le code suivant:

```
<!--[if lt IE 9]>
<script
   src="respond.min.js">
</script>
<![endif]-->
```

CSS Mediaqueries est une autre bibliothèque qui fait la même chose. Le code pour ajouter cette bibliothèque à votre code HTML serait identique:

```
<!--[if lt IE 9]>
<script
   src="css3-mediaqueries.js">
</script>
<![endif]-->
```

L'alternative

Si vous n'aimez pas une solution basée sur JS, vous devriez également envisager d'ajouter une feuille de style IE <9 uniquement, dans laquelle vous ajusterez votre style spécifique à IE <9. Pour cela, vous devez ajouter le code HTML suivant à votre code:

```
<!--[if lt IE 9]>
<link rel="stylesheet" type="text/css" media="all" href="style-ielt9.css"/>
<![endif]-->
```

Remarque :

Techniquement, c'est une alternative: utiliser les **hacks CSS** pour cibler IE <9. Il a le même impact qu'une feuille de style IE <9 uniquement, mais vous n'avez pas besoin d'une feuille de style séparée pour cela. Je ne recommande pas cette option, car ils produisent du code CSS invalide (ce qui n'est que l'une des nombreuses raisons pour lesquelles l'utilisation des hacks CSS est généralement mal vue aujourd'hui).

Lire Requêtes médias en ligne: https://riptutorial.com/fr/css/topic/317/requetes-medias

Chapitre 45: Requêtes sur les fonctionnalités

Syntaxe

- @supports [condition] {/ * Règles CSS à appliquer * /}

Paramètres

Paramètre	Détails
`(property: value)`	Évalue true si le navigateur peut gérer la règle CSS. La parenthèse autour de la règle est requise.
`and`	Renvoie true uniquement si les conditions précédentes et suivantes sont vraies.
`not`	Annule la condition suivante
`or`	Renvoie true si la condition précédente ou suivante est vraie.
`(...)`	Conditions des groupes

Remarques

La détection des fonctionnalités à l'aide de `@supports` est prise en charge dans Edge, Chrome, Firefox, Opera et Safari 9 et `@supports` ultérieures.

Examples

Utilisation de base @supports

```
@supports (display: flex) {
  /* Flexbox is available, so use it */
  .my-container {
    display: flex;
  }
}
```

En termes de syntaxe, `@supports` est très similaire à `@media` , mais au lieu de détecter la taille et l'orientation de l'écran, `@supports` détecte si le navigateur peut gérer une règle CSS donnée.

Plutôt que de faire quelque chose comme `@supports (flex)` , notez que la règle est `@supports (display: flex)` .

Détection des fonctions de chaînage

Pour détecter plusieurs fonctionnalités à la fois, utilisez l'opérateur `and` .

```
@supports (transform: translateZ(1px)) and (transform-style: preserve-3d) and (perspective:
1px) {
  /* Probably do some fancy 3d stuff here */
}
```

Il y a aussi un opérateur `or` un opérateur et un opérateur `not` :

```
@supports (display: flex) or (display: table-cell) {
  /* Will be used if the browser supports flexbox or display: table-cell */
}
@supports not (-webkit-transform: translate(0, 0, 0)) {
  /* Will *not* be used if the browser supports -webkit-transform: translate(...) */
}
```

Pour l'expérience ultime `@supports` , essayez de regrouper les expressions logiques avec des parenthèses:

```
@supports ((display: block) and (zoom: 1)) or ((display: flex) and (not (display: table-
cell))) or (transform: translateX(1px)) {
  /* ... */
}
```

Cela fonctionnera si le navigateur

1. Supporte l' `display: block` ET `zoom: 1` ou
2. Prend en charge l' `display: flex` ET pas `display: table-cell` , ou
3. Prend en charge la `transform: translateX(1px)` .

Lire Requêtes sur les fonctionnalités en ligne: https://riptutorial.com/fr/css/topic/5024/requetes-sur-les-fonctionnalites

Chapitre 46: Sélecteurs

Introduction

Les sélecteurs CSS identifient des éléments HTML spécifiques en tant que cibles pour les styles CSS. Cette rubrique traite de la manière dont les sélecteurs CSS ciblent les éléments HTML. Les sélecteurs utilisent un large éventail de plus de 50 méthodes de sélection proposées par le langage CSS, notamment des éléments, des classes, des identifiants, des pseudo-éléments et des pseudo-classes et des modèles.

Syntaxe

- # *id*
- . *nom du cours*
- : *pseudo-classname*
- :: *pseudo-elementname*
- [*attr*] / * possède l'attribut *attr* . * /
- [*attr* = " *value* "] / * possède l'attribut *attr* et sa valeur est exactement " *value* ". * /
- [*attr* ~ = " *value* "] / * possède l'attribut *attr* et sa valeur, lorsqu'elle est divisée sur des **espaces** , contient " *value* ". * /
- [*attr* | = " *value* "] / * possède l'attribut *attr* et sa valeur est exactement " *value* " **ou** sa valeur commence par " *value* **-** ". * /
- [*attr* ^ = " *value* "] / * possède l'attribut *attr* et sa valeur commence par " *value* ". * /
- [*attr* $ = " *value* "] / * possède l'attribut *attr* et sa valeur se termine par " *value* ". * /
- [*attr* * = " *value* "] / * possède l'attribut *attr* et sa valeur contient " *value* ". * /
- *nom de l'élément*
- *

Remarques

- Parfois , vous verrez doubles points (`::`) au lieu d'un seul (`:`). C'est un moyen de séparer les pseudo-classes des pseudo-éléments .
- Les anciens navigateurs, comme Internet Explorer 8, prennent en charge un **seul** colon (`:`) pour définir les pseudo-éléments.
- Contrairement aux pseudo-classes, un seul pseudo-élément peut être utilisé par sélecteur, s'il est présent, il doit apparaître après la séquence de sélecteurs simples représentant les sujets du sélecteur (une future version de la spécification W3C pourra autoriser plusieurs pseudo-éléments par sélecteur)).

Examples

Sélecteurs d'attribut

Vue d'ensemble

Les sélecteurs d'attribut peuvent être utilisés avec différents types d'opérateurs qui modifient les critères de sélection en conséquence. Ils sélectionnent un élément en utilisant la présence d'un attribut ou d'une valeur d'attribut donné.

Sélecteur [1]	Élément assorti	Sélectionne des éléments ...	Version CSS
`[attr]`	`<div attr>`	Avec attribut `attr`	2
`[attr='val']`	`<div attr="val">`	Où l'attribut `attr` a la valeur `val`	2
`[attr~='val']`	`<div attr="val val2 val3">`	Où `val` apparaît dans le liste séparés par des espaces de `attr`	2
`[attr^='val']`	`<div attr="val1 val2">`	Où la valeur d' `attr` *commence* par `val`	3
`[attr$='val']`	`<div attr="sth aval">`	Où la valeur d' `attr` *se termine* par `val`	3
`[attr*='val']`	`<div attr="somevalhere">`	Où `attr` contient `val` partout	3
`[attr|='val']`	`<div attr="val-sth etc">`	Où la valeur de `attr` est exactement `val` , ou commence par `val` et immédiatement suivi de `-` (U + 002D)	2
`[attr='val' i]`	`<div attr="val">`	Où `attr` a la valeur `val` , ignorer le `val` lettres de `val` .	4 [2]

Remarques:

1. La valeur de l'attribut peut être entourée de guillemets simples ou de guillemets doubles. Aucune citation ne peut également fonctionner, mais elle n'est pas valide selon le standard CSS et elle est déconseillée.

2. Il n'y a pas de spécification CSS4 unique et intégrée, car elle est divisée en modules distincts. Cependant, il existe des modules de "niveau 4". Voir le support du navigateur .

Détails

`[attribute]`

Sélectionne des éléments avec l'attribut donné.

```
div[data-color] {
  color: red;
}
```

```
<div data-color="red">This will be red</div>
<div data-color="green">This will be red</div>
<div data-background="red">This will NOT be red</div>
```

Démo en direct sur JSBin

`[attribute="value"]`

Sélectionne des éléments avec l'attribut et la valeur donnés.

```
div[data-color="red"] {
  color: red;
}
```

```
<div data-color="red">This will be red</div>
<div data-color="green">This will NOT be red</div>
<div data-color="blue">This will NOT be red</div>
```

Démo en direct sur JSBin

`[attribute*="value"]`

Sélectionne des éléments avec l'attribut et la valeur donnés où l'attribut donné contient la valeur donnée n'importe où (sous la forme d'une sous-chaîne).

```
[class*="foo"] {
  color: red;
}
```

```
<div class="foo-123">This will be red</div>
<div class="foo123">This will be red</div>
<div class="bar123foo">This will be red</div>
<div class="barfooo123">This will be red</div>
<div class="barfo0">This will NOT be red</div>
```

Démo en direct sur JSBin

`[attribute~="value"]`

Sélectionne des éléments avec l'attribut et la valeur donnés où la valeur donnée apparaît dans une liste séparée par des espaces.

```
[class~="color-red"] {
  color: red;
}
```

```
<div class="color-red foo-bar the-div">This will be red</div>
<div class="color-blue foo-bar the-div">This will NOT be red</div>
```

Démo en direct sur JSBin

[attribute^="value"]

Sélectionne des éléments avec l'attribut et la valeur donnés où l'attribut donné commence par la valeur.

```
[class^="foo-"] {
  color: red;
}
```

```
<div class="foo-123">This will be red</div>
<div class="foo-234">This will be red</div>
<div class="bar-123">This will NOT be red</div>
```

Démo en direct sur JSBin

[attribute$="value"]

Sélectionne des éléments avec l'attribut et la valeur donnés où l'attribut donné se termine par la valeur donnée.

```
[class$="file"] {
  color: red;
}
```

```
<div class="foobar-file">This will be red</div>
<div class="foobar-file">This will be red</div>
<div class="foobar-input">This will NOT be red</div>
```

Démo en direct sur JSBin

[attribute|="value"]

Sélectionne des éléments avec un attribut et une valeur donnés où la valeur de l'attribut correspond exactement à la valeur donnée ou correspond exactement à la valeur donnée suivie de - (U + 002D)

```
[lang|="EN"] {
  color: red;
}
```

```
<div lang="EN-us">This will be red</div>
<div lang="EN-gb">This will be red</div>
<div lang="PT-pt">This will NOT be red</div>
```

Démo en direct sur JSBin

`[attribute="value" i]`

Sélectionne des éléments avec un attribut et une valeur donnés, où la valeur de l'attribut peut être représentée par `Value` , `VALUE` , `vAlUe` ou toute autre possibilité insensible à la casse.

```
[lang="EN" i] {
  color: red;
}
```

```
<div lang="EN">This will be red</div>
<div lang="en">This will be red</div>
<div lang="PT">This will NOT be red</div>
```

Démo en direct sur JSBin

Spécificité des sélecteurs d'attribut

0-1-0

Identique au sélecteur de classe et à la pseudo-classe.

```
*[type=checkbox] // 0-1-0
```

Notez que cela signifie qu'un sélecteur d'attribut peut être utilisé pour sélectionner un élément par son ID à un niveau de spécificité inférieur à celui s'il était sélectionné avec un sélecteur d'ID : `[id="my-ID"]` cible le même élément que `#my-ID` mais avec une spécificité inférieure.

Voir la section Syntaxe pour plus de détails.

Combinateurs

Vue d'ensemble

Sélecteur	La description
`div span`	Sélecteur de descendant (tous les `<span>` s qui sont les descendants d'un `<div>`)
`div > span`	Sélecteur d'enfants (tous les `<span>` enfants directs d'un `<div>`)
`a ~ span`	Sélecteur général de frères et sœurs (tous les `<span>` frères et sœurs après un `<a>`)
`a + span`	Sélecteur de frères et sœurs adjacent (tous les `<span>` s qui sont immédiatement après un `<a>`)

Remarque: Les sélecteurs frères et sœurs ciblent les éléments qui les suivent dans le

document source. CSS, de par sa nature (il cascade), ne peut pas cibler *les* éléments *précédents* ou *parents* . Cependant, en utilisant la propriété flex `order` , un sélecteur précédent peut être simulé sur un support visuel .

Combinateur descendant: `selector selector`

Un combinateur descendant, représenté par au moins un caractère d'espace ( ), sélectionne les éléments descendants de l'élément défini. Ce combinateur sélectionne **tous les** descendants de l'élément (des éléments enfants dessus).

```
div p {
  color:red;
}
```

```
<div>
  <p>My text is red</p>
  <section>
    <p>My text is red</p>
  </section>
</div>

<p>My text is not red</p>
```

Démo en direct sur JSBin

Dans l'exemple ci-dessus, les deux premiers éléments `<p>` sont sélectionnés car ils sont tous deux des descendants de `<div>` .

Combinateur d'enfants: `selector > selector`

Le combinateur enfant (`>`) est utilisé pour sélectionner des éléments qui sont des **enfants** ou **des descendants directs** de l'élément spécifié.

```
div > p {
  color:red;
}
```

```
<div>
  <p>My text is red</p>
  <section>
    <p>My text is not red</p>
  </section>
</div>
```

Démo en direct sur JSBin

Le CSS ci-dessus sélectionne uniquement le premier élément `<p>` , car c'est le seul paragraphe directement issu d'un `<div>` .

Le deuxième élément `<p>` n'est pas sélectionné car il ne s'agit pas d'un enfant direct de `<div>` .

Combinateur fraternel adjacent: `selector + selector`

Le combinateur frère (`+`) adjacent sélectionne un élément frère qui suit immédiatement un élément spécifié.

```
p + p {
  color:red;
}
```

```
<p>My text is not red</p>
<p>My text is red</p>
<p>My text is red</p>
<hr>
<p>My text is not red</p>
```

Démo en direct sur JSBin

L'exemple ci-dessus sélectionne uniquement les éléments `<p>` qui sont *directement précédés* par un autre élément `<p>` .

Combinateur général de frères et sœurs: `selector ~ selector`

Le combinateur général de frères et sœurs (`~`) sélectionne *tous les* frères qui suivent l'élément spécifié.

```
p ~ p {
  color:red;
}
```

```
<p>My text is not red</p>
<p>My text is red</p>
<hr>
<h1>And now a title</h1>
<p>My text is red</p>
```

Démo en direct sur JSBin

L'exemple ci-dessus sélectionne tous les éléments `<p>` *précédés* d'un autre élément `<p>` , qu'ils soient ou non immédiatement adjacents.

Sélecteurs de nom de classe

Le sélecteur de nom de classe sélectionne tous les éléments avec le nom de la classe ciblée. Par

exemple, le nom de classe `.warning` sélectionnera l'élément `<div>` :

```
<div class="warning">
    <p>This would be some warning copy.</p>
</div>
```

Vous pouvez également combiner des noms de classe pour cibler des éléments plus spécifiquement. Prenons l'exemple ci-dessus pour présenter une sélection de classes plus compliquée.

CSS

```
.important {
    color: orange;
}
.warning {
    color: blue;
}
.warning.important {
    color: red;
}
```

HTML

```
<div class="warning">
    <p>This would be some warning copy.</p>
</div>

<div class="important warning">
    <p class="important">This is some really important warning copy.</p>
</div>
```

Dans cet exemple, tous les éléments de la `.warning` classe auront une couleur de texte en bleu, des éléments avec la `.important` classe ont une couleur de texte orange, et tous les éléments qui ont à la *fois* le `.important` et `.warning` nom de la classe auront un texte rouge Couleur.

Notez que dans la CSS, la déclaration `.warning.important` ne comportait aucun espace entre les deux noms de classe. Cela signifie qu'il ne trouvera des éléments qui contiennent les noms de classe `warning` et `important` dans leur `class` attribut. Ces noms de classe peuvent être dans n'importe quel ordre sur l'élément.

Si un espace était inclus entre les deux classes dans la déclaration CSS, il ne sélectionnerait que les éléments ayant des éléments parents avec un `.warning` classe `.warning` et des éléments enfants avec des noms de classe `.important` .

Sélecteurs d'ID

Les sélecteurs d'ID sélectionnent les éléments DOM avec l'ID ciblé. Pour sélectionner un élément par un identifiant spécifique en CSS, le préfixe `#` est utilisé.

Par exemple, l'élément `div` suivant HTML…

```
<div id="exampleID">
    <p>Example</p>
</div>
```

… Peut être sélectionné par `#exampleID` en CSS comme indiqué ci-dessous:

```
#exampleID {
    width: 20px;
}
```

> **Remarque** : les spécifications HTML n'autorisent pas plusieurs éléments avec le même ID

Les pseudo-classes

Les pseudo-classes sont des **mots** - **clés** qui permettent une sélection basée sur des informations situées en dehors de l'arbre des documents ou qui ne peuvent pas être exprimées par d'autres sélecteurs ou combinateurs. Ces informations peuvent être associées à un certain état (pseudo-classes d' état et dynamiques), à des emplacements (pseudo-classes structurelles et cibles), à des négations de la première (pseudo-classe de négation) ou à des langages (pseudo-classe lang). Les exemples incluent si oui ou non un lien a été suivi (`:visited`), la souris survole un élément (`:hover`), une case à cocher est cochée (`:checked`), etc.

Syntaxe

```
selector:pseudo-class {
    property: value;
}
```

Liste des pseudo-classes:

prénom	La description
`:active`	S'applique à tout élément activé (c.-à-d. Cliqué) par l'utilisateur.
`:any`	Vous permet de créer des ensembles de sélecteurs associés en créant des groupes les articles inclus correspondront. Ceci est une alternative à la répétition d'un sélecteur entier.
`:target`	Sélectionne l'élément #news actif actuel (cliqué sur une URL contenant ce nom d'ancrage)
`:checked`	S'applique à la radio, aux cases à cocher ou aux éléments d'option cochés ou basculé dans un état "on".
`:default`	Représente tout élément d'interface utilisateur par défaut dans un groupe

prénom	La description
	de éléments similaires.
`:disabled`	S'applique à tout élément d'interface utilisateur qui est désactivé.
`:empty`	S'applique à tout élément qui n'a pas d'enfants.
`:enabled`	S'applique à tout élément d'interface utilisateur qui est activé.
`:first`	Utilisé conjointement avec la règle `@page` , cela sélectionne la première page d'un document imprimé.
`:first-child`	Représente tout élément qui est le premier élément enfant de son parent.
`:first-of-type`	S'applique lorsqu'un élément est le premier du type d'élément sélectionné à l'intérieur de son parent. Cela peut ou peut ne pas être le premier enfant.
`:focus`	S'applique à tout élément ayant le focus de l'utilisateur. Cela peut être donné par le clavier de l'utilisateur, événements de la souris ou autres formes de saisie.
`:focus-within`	Peut être utilisé pour mettre en évidence une section entière lorsqu'un élément à l'intérieur est focalisé. Il correspond à tout élément correspondant à la pseudo-classe: focus ou ayant un descendant ciblé.
`:full-screen`	S'applique à tout élément affiché en mode plein écran. Il sélectionne toute la pile des éléments et pas seulement l'élément de niveau supérieur.
`:hover`	S'applique à tout élément survolé par le périphérique de pointage de l'utilisateur, mais non activé.
`:indeterminate`	Applique des éléments d'interface radio ou de case à cocher qui ne sont ni cochés ni non cochée, mais sont dans un état indéterminé. Cela peut être dû à un l'attribut d'élément ou la manipulation de DOM.
`:in-range`	La pseudo-classe CSS `:in-range` correspond à un élément son attribut value dans les limites de plage spécifiées pour cet élément. Il permet à la page de donner un retour que la valeur actuellement définie L'utilisation de l'élément se situe dans les limites de la plage.
`:invalid`	S'applique aux éléments `<input>` dont les valeurs sont invalides selon le type spécifié dans l'attribut `type=` .
`:lang`	S'applique à tout élément qui contient un élément `<body>` enveloppé

prénom	La description
	correctement désigné `lang=` attribut. Pour que la pseudo-classe soit valide, elle doit contient un code de langue valide à deux ou trois lettres.
`:last-child`	Représente tout élément qui est le dernier élément enfant de son parent.
`:last-of-type`	S'applique lorsqu'un élément est le dernier du type d'élément sélectionné à l'intérieur son parent. Cela peut ou peut ne pas être le dernier enfant.
`:left`	Utilisé conjointement avec la règle `@page` , cela sélectionne tout le côté gauche pages dans un document imprimé.
`:link`	S'applique à tous les liens qui n'ont pas été visités par l'utilisateur.
`:not()`	S'applique à tous les éléments qui **ne** correspondent **pas** à la valeur transmise à (`:not(p)` ou `:not(.class-name)` par exemple. Il doit avoir une valeur pour être valide et il ne peut contenir qu'un seul sélecteur. Cependant, vous pouvez enchaîner plusieurs `:not` sélecteurs ensemble.
`:nth-child`	Applique quand un élément est le `n` élément de -ième de son parent, où `n` peut être un entier, une expression mathématique (par exemple `n+3`) ou les mots-clés `odd` ou `even` .
`:nth-of-type`	Applique quand un élément est le `n` élément -ième de son parent du même type d'élément, où `n` peut être un entier, un mathématique expression (par exemple `n+3`) ou les mots clés `odd` ou `even` .
`:only-child`	La pseudo-classe CSS `:only-child` représente un élément quelconque qui est le seul enfant de son parent. C'est la même chose que `:first-child:last-child` ou `:nth-child(1):nth-last-child(1)` , mais avec une spécificité inférieure.
`:optional`	La pseudo-classe CSS `:optional` représente n'importe quel élément qui ne comporte pas l'attribut requis. Ceci permet des formulaires pour indiquer facilement les champs facultatifs et les styliser en conséquence.
`:out-of-range`	Le `:out-of-range` pseudo-classe CSS `:out-of-range` correspond à un élément attribut value en dehors des limites de plage spécifiées pour cet élément. Il permet à la page de donner un retour que la valeur actuellement définie en utilisant le

prénom	La description
	l'élément est en dehors des limites de la plage. Une valeur peut être en dehors d'une plage si elle est soit plus petit ou plus grand que les valeurs maximales et minimales définies.
`:placeholder-shown`	**Expérimental.** S'applique à tout élément de formulaire affichant actuellement un texte d'espace réservé.
`:read-only`	S'applique à tout élément non modifiable par l'utilisateur.
`:read-write`	S'applique à tout élément modifiable par un utilisateur, tel que `<input>` éléments `<input>` .
`:right`	Utilisé en conjonction avec la règle `@page` , cela sélectionne toutes les bonnes pages dans un document imprimé.
`:root`	correspond à l'élément racine d'une arborescence représentant le document.
`:scope`	La pseudo-classe CSS correspond aux éléments qui sont une référence point pour que les sélecteurs correspondent.
`:target`	Sélectionne l'élément #news actif actuel (cliqué sur une URL contenant ce nom d'ancrage)
`:visited`	S'applique à tous les liens qui ont été visités par l'utilisateur.

La pseudo-classe `:visited` Visitée ne peut plus être utilisée pour la plupart des styles de navigateurs modernes, car il s'agit d'une faille de sécurité. Voir ce lien pour référence.

Sélecteurs de base

Sélecteur	La description
`*`	Sélecteur universel (tous les éléments)
`div`	Sélecteur de tag (tous les éléments `<div>`)
`.blue`	Sélecteur de classe (tous les éléments avec la classe `blue`)
`.blue.red`	Tous les éléments avec la classe `blue` **et** `red` (un type de sélecteur composé)
`#headline`	Sélecteur d'ID (l'élément avec l'attribut "id" défini sur le `headline`)

Sélecteur	La description
`:pseudo-class`	Tous les éléments avec pseudo-classe
`::pseudo-element`	Elément correspondant au pseudo-élément
`:lang(en)`	Elément correspondant à: déclaration lang, par exemple `<span lang="en">`
`div > p`	sélecteur enfant

> **Remarque:** la valeur d'un ID doit être unique dans une page Web. Utiliser la valeur d'un ID plus d'une fois dans le même arbre de document constitue une violation du standard HTML .

Une liste complète des sélecteurs se trouve dans la spécification CSS Selectors Level 3 .

Comment styler une entrée de plage

HTML

```
<input type="range"></input>
```

CSS

Effet	Pseudo sélecteur
Pouce	`input[type=range]::-webkit-slider-thumb, input[type=range]::-moz-range-thumb, input[type=range]::-ms-thumb`
Piste	`input[type=range]::-webkit-slider-runnable-track, input[type=range]::-moz-range-track, input[type=range]::-ms-track`
OnFocus	`input[type=range]:focus`
Partie inférieure de la piste	`input[type=range]::-moz-range-progress, input[type=range]::-ms-fill-lower` (impossible dans les navigateurs WebKit actuellement - JS nécessaire)

Valeur booléenne globale avec case à cocher: cochée et ~ (combinateur frère général)

Avec le sélecteur ~, vous pouvez facilement implémenter un booléen accessible global sans utiliser JavaScript.

Ajouter une valeur booléenne comme case à cocher

Au tout début de votre document, ajoutez autant de booléens que vous voulez avec un `id` unique et le jeu d'attributs `hidden` :

```
<input type="checkbox" id="sidebarShown" hidden />
<input type="checkbox" id="darkThemeUsed" hidden />

<!-- here begins actual content, for example: -->
<div id="container">
    <div id="sidebar">
        <!-- Menu, Search, ... -->
    </div>

    <!-- Some more content ... -->
</div>

<div id="footer">
    <!-- ... -->
</div>
```

Changer la valeur du booléen

Vous pouvez basculer le booléen en ajoutant une `label` avec l'attribut `for` :

```
<label for="sidebarShown">Show/Hide the sidebar!</label>
```

Accéder à la valeur booléenne avec CSS

Le sélecteur normal (comme `.color-red`) spécifie les propriétés par défaut. Ils peuvent être remplacés en suivant les sélecteurs `true` / `false` :

```
/* true: */
<checkbox>:checked ~ [sibling of checkbox & parent of target] <target>

/* false: */
<checkbox>:not(:checked) ~ [sibling of checkbox & parent of target] <target>
```

Notez que `<checkbox>` , `[sibling ...]` et `<target>` doivent être remplacés par les sélecteurs appropriés. `[sibling ...]` peut être un sélecteur spécifique (souvent si vous êtes paresseux) simplement `*` ou rien si la cible est déjà un frère de la case à cocher.

Des exemples pour la structure HTML ci-dessus seraient:

```
#sidebarShown:checked ~ #container #sidebar {
    margin-left: 300px;
}

#darkThemeUsed:checked ~ #container,
#darkThemeUsed:checked ~ #footer {
    background: #333;
}
```

En action

Voir ce violon pour une implémentation de ces booléens globaux.

CSS3: exemple de sélecteur in-range

```
<style>
input:in-range {
    border: 1px solid blue;
}
</style>



<input type="number" min="10" max="20" value="15">
<p>The border for this value will be blue</p>
```

La pseudo-classe CSS `:in-range` correspond lorsqu'un élément a son attribut value dans les limites de plage spécifiées pour cet élément. Cela permet à la page de donner un avis indiquant que la valeur actuellement définie à l'aide de l'élément est comprise dans les limites de la plage. [1]

Classe de pseudo enfant

> "La pseudo-classe CSS nth-child (an + b) correspond à un élément qui contient un frère + + b-1 dans l'arbre de document, pour une **valeur** positive **ou nulle** donnée pour n" - MDN: nth-child

pseudo-sélecteur	1	2	3	4	5	6	7	8	9	dix
`:first-child`	✔									
`:nth-child(3)`			✔							
`:nth-child(n+3)`			✔	✔	✔	✔	✔	✔	✔	✔
`:nth-child(3n)`			✔			✔			✔	
`:nth-child(3n+1)`	✔			✔			✔			✔
`:nth-child(-n+3)`	✔	✔	✔							
`:nth-child(odd)`	✔		✔		✔		✔		✔	
`:nth-child(even)`		✔		✔		✔		✔		✔
`:last-child`										✔

pseudo-sélecteur	1	2	3	4	5	6	7	8	9	dix
`:nth-last-child(3)`								✔		

Sélectionner l'élément en utilisant son identifiant sans la haute spécificité du sélecteur d'identifiant

Cette astuce vous permet de sélectionner un élément utilisant l'ID comme valeur pour un sélecteur d'attribut afin d'éviter la grande spécificité du sélecteur d'ID.

HTML:

```
<div id="element">...</div>
```

CSS

```
#element { ... } /* High specificity will override many selectors */

[id="element"] { ... } /* Low specificity, can be overridden easily */
```

A. L'exemple: non pseudo-classe & B.: pseudo-classe CSS focus-within

A. La syntaxe est présentée ci-dessus.

Le sélecteur suivant correspond à tous `<input>` éléments `<input>` d'un document HTML qui ne sont pas désactivés et n'ont pas la classe `.example` :

HTML:

```
<form>
    Phone: <input type="tel" class="example">
    E-mail: <input type="email" disabled="disabled">
    Password: <input type="password">
</form>
```

CSS:

```
input:not([disabled]):not(.example){
   background-color: #ccc;
}
```

La pseudo-classe `:not()` supportera également les sélecteurs séparés par des virgules dans les sélecteurs de niveau 4:

CSS:

```
input:not([disabled], .example){
   background-color: #ccc;
}
```

Démo en direct sur JSBin

Voir la syntaxe de fond ici .

B. La pseudo-classe CSS focus-within

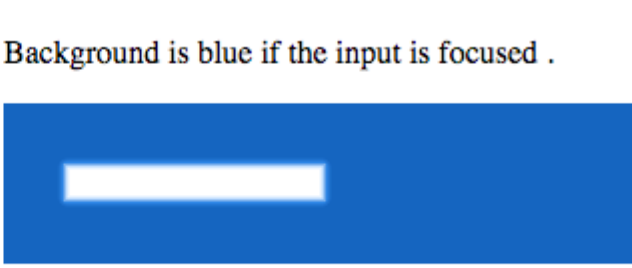
HTML:

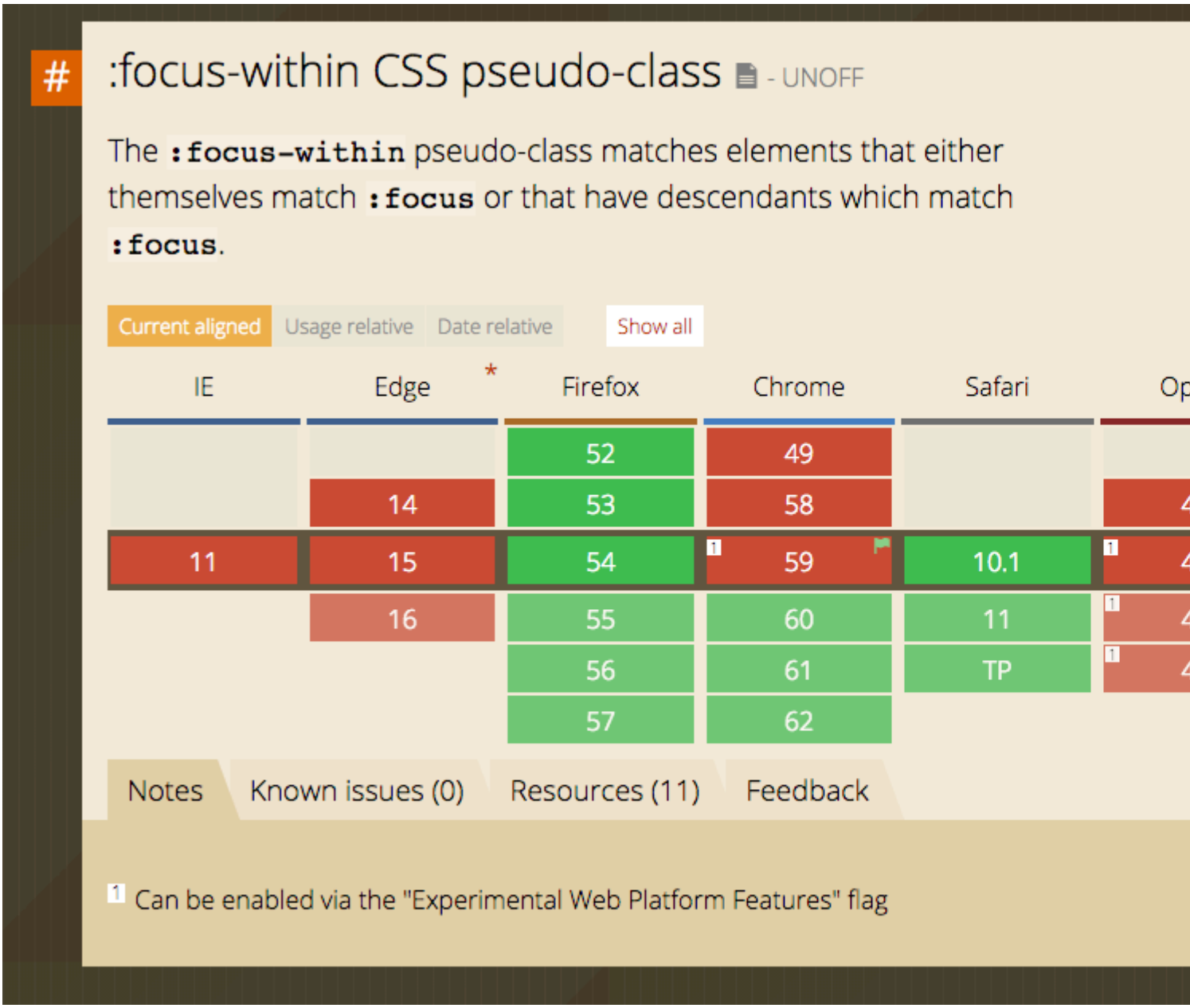
```
  <h3>Background is blue if the input is focused .</p>
  <div>
    <input type="text">
  </div>
```

CSS:

```
div {
  height: 80px;
}
input{
  margin:30px;
}
div:focus-within {
  background-color: #1565C0;
}
```

```
div {
  height: 80px;
}
input{
  margin:30px;
}
div:focus-within {
  background-color: #1565C0;
}
```





Exemple de sélecteur de pseudo-classe: only-child

La pseudo-classe `CSS :only-child` représente tout élément qui est le seul enfant de son parent.

HTML:

```
<div>
  <p>This paragraph is the only child of the div, it will have the color blue</p>
</div>

<div>
  <p>This paragraph is one of the two children of the div</p>
  <p>This paragraph is one of the two children of its parent</p>
</div>
```

CSS:

```
p:only-child {
```

```
  color: blue;
}
```

L'exemple ci-dessus sélectionne l'élément `<p>` qui est l'enfant unique de son parent, en l'occurrence un `<div>` .

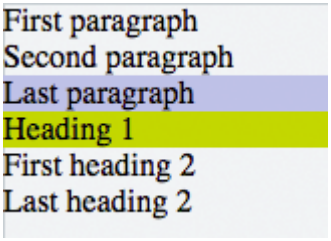
Démo en direct sur JSBin

Le: sélecteur dernier type

Le `:last-of-type` sélectionne l'élément qui est le dernier enfant, d'un type particulier, de son parent. Dans l'exemple ci-dessous, le css sélectionne le dernier paragraphe et le dernier en-tête `h1` .

```
p:last-of-type {
  background: #C5CAE9;
}
h1:last-of-type {
  background: #CDDC39;
}
```

```
<div class="container">
    <p>First paragraph</p>
    <p>Second paragraph</p>
    <p>Last paragraph</p>
    <h1>Heading 1</h1>
    <h2>First heading 2</h2>
    <h2>Last heading 2</h2>
</div>
```



jsfiddle

Lire Sélecteurs en ligne: https://riptutorial.com/fr/css/topic/611/selecteurs

Chapitre 47: Sprites d'image CSS

Syntaxe

- **// Utilisation de la position d'arrière-plan**
  background: url ("sprite-image.png");
  position de fond: -20px 50px;
- **// raccourci de propriété d'arrière-plan**
  background: url ("sprite-image.png") -20px 50px;

Remarques

Pour certains cas d'utilisation, les sprites sont de moins en moins utilisés, remplacés par des icônes Web ou des images SVG .

Examples

Une implémentation de base

Qu'est-ce qu'un sprite d'image?

Un sprite d'image est un actif unique situé dans une feuille d'image-objet. Une feuille d'image-objet image est un fichier image contenant plus d'un élément pouvant être extrait.

Par exemple:

L'image ci-dessus est une feuille d'image-objet et chacune de ces étoiles est une image-objet dans la feuille de sprite. Ces feuilles de sprite sont utiles car elles améliorent les performances en réduisant le nombre de requêtes HTTP qu'un navigateur doit effectuer.

Alors, comment en implémentez-vous un? Voici un exemple de code.

HTML

```
<div class="icon icon1"></div>
<div class="icon icon2"></div>
```

```
<div class="icon icon3"></div>
```

CSS

```
.icon {
    background: url(“icons-sprite.png”);
    display: inline-block;
    height: 20px;
    width: 20px;
}
.icon1 {
      background-position: 0px 0px;
}
.icon2 {
      background-position: -20px 0px;
}
.icon3 {
      background-position: -40px 0px;
}
```

En définissant la largeur et la hauteur de l'image-objet et en utilisant la propriété background-position en CSS (avec une valeur x et y), vous pouvez facilement extraire des images-objets à partir d'une feuille de sprite à l'aide de CSS.

Lire Sprites d'image CSS en ligne: https://riptutorial.com/fr/css/topic/3690/sprites-d-image-css

Chapitre 48: Structure et mise en forme d'une règle CSS

Remarques

Pour faciliter la lecture, conservez toutes les déclarations en retrait d'un niveau par rapport à leur sélecteur et les accolades fermantes sur leur propre ligne. Ajoutez un seul espace après les sélecteurs et les deux-points, et placez toujours un point-virgule après la déclaration finale.

Bien

```
p {
    color: maroon;
    font-size: 16px;
}
```

Mal

```
p{
 color: maroon;
font-size:16px }
```

Bon mot

S'il n'y a qu'une ou deux déclarations, vous *pourriez* vous en sortir avec celle-ci. Non recommandé pour la plupart des cas. Toujours être cohérent lorsque cela est possible.

```
p { color: maroon; font-size: 16px; }
```

Examples

Règles, sélecteurs et blocs de déclaration

Une **règle** CSS consiste en un **sélecteur** (par exemple, `h1`) et un **bloc de déclaration** (`{}`).

```
h1 {}
```

Listes de propriété

Certaines propriétés peuvent prendre plusieurs valeurs, appelées collectivement une **liste de**

propriétés .

```
/* Two values in this property list */
span {
    text-shadow: yellow 0 0 3px, green 4px 4px 10px;
}

/* Alternate Formatting */
span {
    text-shadow:
        yellow 0 0 3px,
        green 4px 4px 10px;
}
```

Sélecteurs multiples

Lorsque vous groupez des sélecteurs CSS, vous appliquez les mêmes styles à plusieurs éléments différents sans répéter les styles de votre feuille de style. Utilisez une virgule pour séparer plusieurs sélecteurs groupés.

```
div, p { color: blue }
```

La couleur bleue s'applique donc à tous les éléments `<div>` et à tous les éléments `<p>` . Sans la virgule, seuls les éléments `<p>` enfants d'un `<div>` seraient rouges.

Cela s'applique également à tous les types de sélecteurs.

```
p, .blue, #first, div span{ color : blue }
```

Cette règle s'applique à:

- `<p>`
- éléments de la classe `blue`
- élément avec l'ID d' `first`
- chaque `<span>` intérieur d'un `<div>`

Lire Structure et mise en forme d'une règle CSS en ligne:
https://riptutorial.com/fr/css/topic/4313/structure-et-mise-en-forme-d-une-regle-css

Chapitre 49: Style de curseur

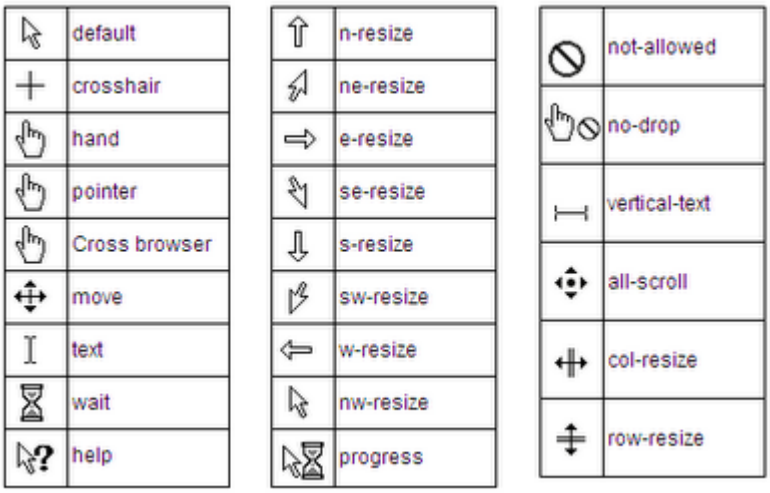
Syntaxe

- curseur: auto | par défaut | aucun menu contextuel | aide | pointeur | progrès | attendre | cellule | croix | texte | texte vertical | alias | copie | déplacer | no-drop | non autorisé | e-redimensionner | n-redimensionner | ne-redimensionner | nw-redimensionner | s-redimensionner | se redimensionner | sw-redimensionner | w-redimensionner | redimensionner | ns-redimensionner | nesw-redimensionner | nwse-redimensionner | redimensionner | redimensionnement des lignes | tout faire défiler | zoom avant | zoom arrière | saisir | saisir;

Examples

Changer le type de curseur

```
cursor: value;
```



Exemples:

Valeur	La description
aucun	Aucun curseur n'est rendu pour l'élément
auto	Défaut. Le navigateur définit un curseur
Aidez-moi	Le curseur indique que l'aide est disponible
attendez	Le curseur indique que le programme est occupé
bouge toi	Le curseur indique que quelque chose doit être déplacé

Valeur	La description
aiguille	Le curseur est un pointeur et indique un lien

pointeur-événements

La propriété pointer-events permet de contrôler la manière dont les éléments HTML répondent aux événements souris / touch.

```
.disabled {
  pointer-events: none;
}
```

Dans cet exemple,

> 'none' empêche toutes les options de clic, d'état et de curseur sur l'élément HTML spécifié [[1]]

Les autres valeurs valides pour les éléments HTMl sont:

- auto;
- hériter.

1. https://css-tricks.com/almanac/properties/p/pointer-events/

Autres ressources:

- https://developer.mozilla.org/en-US/docs/Web/CSS/pointer-events
- https://davidwalsh.name/pointer-events

couleur caret

La propriété CSS caret-color spécifie la couleur du signe d'insertion, l'indicateur visible du point d'insertion dans un élément où du texte et un autre contenu sont insérés lors de la saisie ou de la modification par l'utilisateur.

HTML

```
<input id="example" />
```

CSS

```
#example {
  caret-color: red;
}
```

Ressources:

- https://developer.mozilla.org/en-US/docs/Web/CSS/caret-color

Lire Style de curseur en ligne: https://riptutorial.com/fr/css/topic/1742/style-de-curseur

Chapitre 50: Styles de liste

Syntaxe

- liste-style: list-style-type | position de style de liste | liste-style-image | initiale | hériter;

Paramètres

Valeur	La description
type liste-style	le type de marqueur d'élément de liste.
position de style de liste	spécifie où placer le marqueur
liste-style-image	spécifie le type de marqueur d'élément de liste
initiale	définit cette propriété à sa valeur par défaut
hériter	hérite de cette propriété de son élément parent

Remarques

Bien que `list-style-type` soit en fait une propriété qui s'applique uniquement aux éléments de liste (normalement `<li>`), elle est souvent spécifiée pour la balise list (`<ol>` ou `<ul>`). Dans ce cas, les éléments de liste héritent de la propriété.

Examples

Type de balle ou de numérotation

Spécifique pour les balises `<li>` dans une liste non ordonnée (`<ul>`):

```
list-style: disc;               /* A filled circle (default) */
list-style: circle;             /* A hollow circle */
list-style: square;             /* A filled square */
list-style: '-';                /* any string */
```

Spécifique pour les balises `<li>` dans une liste ordonnée (`<ol>`):

```
list-style: decimal;            /* Decimal numbers beginning with 1 (default) */
list-style: decimal-leading-zero;/* Decimal numbers padded by initial zeros (01, 02, 03, … 10)
*/
list-style: lower-roman;        /* Lowercase roman numerals (i., ii., iii., iv., ...) */
list-style: upper-roman;        /* Uppercase roman numerals (I., II., III., IV., ...) */
list-style-type: lower-greek;   /* Lowercase roman letters (α., β., γ., δ., ...) */
list-style-type: lower-alpha;   /* Lowercase letters (a., b., c., d., ...) */
```

```
list-style-type: lower-latin;    /* Lowercase letters (a., b., c., d., ...) */
list-style-type: upper-alpha;    /* Uppercase letters (A., B., C., D., ...) */
list-style-type: upper-latin;    /* Uppercase letters (A., B., C., D., ...) */
```

Non spécifique:

```
list-style: none;                /* No visible list marker */
list-style: inherit;             /* Inherits from parent */
```

Position de la balle

Une liste est composée d'éléments `<li>` intérieur d'un élément contenant (`<ul>` ou `<ol>`). Les éléments de liste et le conteneur peuvent avoir des marges et des remplissages qui influencent la position exacte du contenu de l'élément de liste dans le document. Les valeurs par défaut pour la marge et le remplissage peuvent être différentes pour chaque navigateur. Afin d'obtenir la même mise en page multi-navigateur, ceux-ci doivent être définis spécifiquement.

Chaque élément de la liste reçoit une "zone de marqueur" contenant le marqueur de puce. Cette case peut être placée à l'intérieur ou à l'extérieur de la zone de liste.

```
list-style-position: inside;
```

place la puce dans l'élément `<li>` , en poussant le contenu vers la droite si nécessaire.

```
list-style-position: outside;
```

place la puce à gauche de l'élément `<li>` . S'il n'y a pas assez d'espace dans le remplissage de l'élément contenant, la zone de marqueur s'étendra vers la gauche, même si elle venait à tomber de la page.

Affichage du résultat du positionnement `inside` et `outside` : jsfiddle

Suppression des puces / numéros

Parfois, une liste ne doit afficher aucun point ou chiffre. Dans ce cas, n'oubliez pas de spécifier la marge et le remplissage.

```
<ul>
    <li>first item</li>
    <li>second item</li>
</ul>
```

CSS

```
ul {
    list-style-type: none;
}
li {
    margin: 0;
```

```
    padding: 0;
}
```

Lire Styles de liste en ligne: https://riptutorial.com/fr/css/topic/4215/styles-de-liste

Chapitre 51: Support du navigateur et préfixes

Paramètres

Préfixe	Navigateur (s)
`-webkit-`	Google Chrome, Safari, nouvelles versions d'Opera 12 et plus, navigateurs Android, Blackberry et UC
`-moz-`	Mozilla Firefox
`-ms-`	Internet Explorer, Edge
`-o-` , `-xv-`	Opera jusqu'à la version 12
`-khtml-`	Konquerer

Remarques

Les préfixes de fournisseur sont utilisés pour permettre la prise en charge de la prévisualisation de nouvelles fonctionnalités CSS dont la fonctionnalité n'est pas encore recommandée par la spécification.

> **Il est recommandé de ne pas utiliser les préfixes de fournisseur dans les environnements de production.** Ces préfixes existent pour tester de nouvelles fonctionnalités qui ne sont pas encore finalisées et le comportement est intrinsèquement inattendu. Il suffit en utilisant des préfixes ne confère **pas** de navigateur pour le soutien des navigateurs anciens que vous ne pouvez pas garantir la fonctionnalité n'a pas changé au fil du temps pour effectuer différemment, et il *pourrait* encore être cassé dans les anciens navigateurs vous prétendez soutenir.
>
> Si la prise en charge d'anciens navigateurs est importante, vous devriez plutôt envisager d'utiliser JavaScript ou d'autres solutions pour imiter les effets et garantir véritablement la prise en charge des anciens navigateurs.

Les navigateurs utiliseront leurs préfixes et ignoreront les propriétés qu'ils ne comprennent pas.

REMARQUE : Les préfixes doivent toujours apparaître avant la syntaxe officielle non préfixée. Sinon, ils seraient remplacés par les propriétés préfixées, ce qui peut être une autre implémentation à la fin.

Si un navigateur prend en charge à la fois une version non préfixée et une version préfixée d'une propriété, la propriété la plus récente à déclarer est prioritaire.

Examples

Transitions

```
div {
    -webkit-transition: all 4s ease;
       -moz-transition: all 4s ease;
         -o-transition: all 4s ease;
            transition: all 4s ease;
}
```

Transformer

```
div {
    -webkit-transform: rotate(45deg);
       -moz-transform: rotate(45deg);
        -ms-transform: rotate(45deg);
         -o-transform: rotate(45deg);
            transform: rotate(45deg);
}
```

Lire Support du navigateur et préfixes en ligne: https://riptutorial.com/fr/css/topic/1138/support-du-navigateur-et-prefixes

Chapitre 52: Transformations 2D

Syntaxe

- **Rotation Transformer**
- transformer: tourner (<angle>)
- **Traduire Transformer**
- transformer: translate (<longueur ou pourcentage> [, <longueur ou pourcentage>]?)
- transformer: translateX (<longueur ou pourcentage>)
- transformer: translateY (<longueur ou pourcentage>)
- **Transformer en biais**
- transform: skew (<angle> [, <angle>]?)
- transformer: skewX (<angle>)
- transformer: skewY (<angle>)
- **Transformation d'échelle**
- transform: scale (<facteur d'échelle> [, <facteur d'échelle>]?)
- transformer: scaleX (<facteur d'échelle>)
- transformer: scaleY (<facteur d'échelle>)
- **Transformation de la matrice**
- transform: matrix (<nombre> [, <nombre>] {5,5})

Paramètres

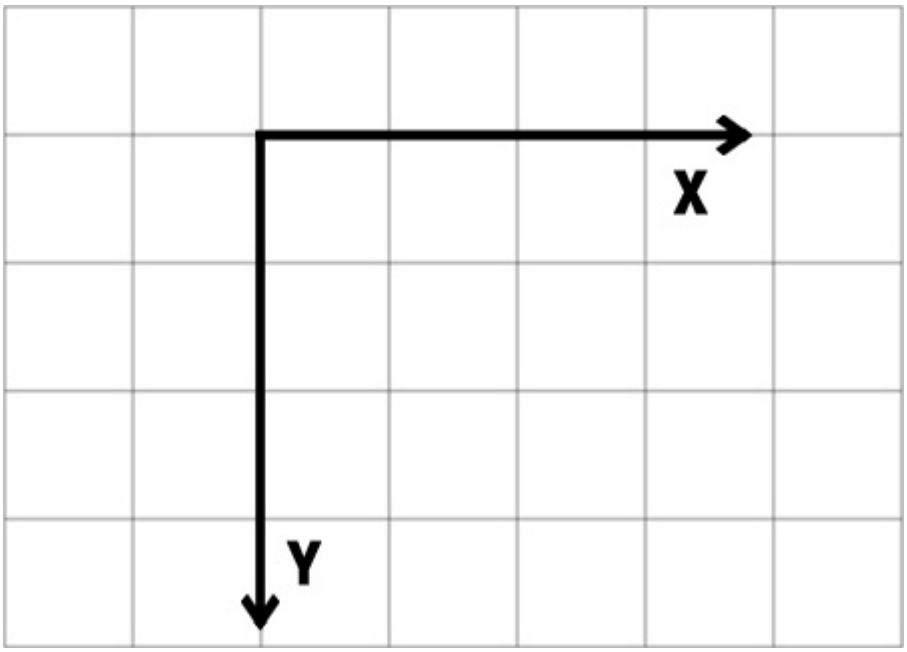
Fonction / Paramètre	Détails
`rotate(x)`	Définit une transformation qui déplace l'élément autour d'un point fixe sur l'axe Z
`translate(x,y)`	Déplace la position de l'élément sur les axes X et Y
`translateX(x)`	Déplace la position de l'élément sur l'axe X
`translateY(y)`	Déplace la position de l'élément sur l'axe Y
`scale(x,y)`	Modifie la taille de l'élément sur les axes X et Y
`scaleX(x)`	Modifie la taille de l'élément sur l'axe X
`scaleY(y)`	Modifie la taille de l'élément sur l'axe Y
`skew(x,y)`	Mappage par cisaillement, ou transvection, déformant chaque point d'un élément d'un certain angle dans chaque direction
`skewX(x)`	Cartographie horizontale du cisaillement déformant chaque point d'un élément d'un certain angle dans la direction horizontale

Fonction / Paramètre	Détails
`skewY(y)`	Cartographie verticale du cisaillement déformant chaque point d'un élément d'un certain angle dans la direction verticale
`matrix()`	Définit une transformation 2D sous la forme d'une matrice de transformation.
angle	L'angle selon lequel l'élément doit être pivoté ou incliné (en fonction de la fonction avec laquelle il est utilisé). L'angle peut être fourni en degrés (`deg`), gradiens (`grad`), radians (`rad`) ou tours (`turn`). Dans la fonction `skew()` , le deuxième angle est facultatif. S'il n'est pas fourni, il n'y aura pas (0) d'inclinaison dans l'axe Y.
longueur ou pourcentage	La distance exprimée en longueur ou en pourcentage par laquelle l'élément doit être traduit. Dans la fonction `translate()` , la deuxième longueur ou pourcentage est facultative. S'il n'est pas fourni, il n'y aurait pas de traduction (0) en ordonnée.
facteur d'échelle	Un nombre qui définit combien de fois l'élément doit être mis à l'échelle dans l'axe spécifié. Dans la fonction `scale()` , le deuxième facteur d'échelle est facultatif. S'il n'est pas fourni, le premier facteur d'échelle sera également appliqué pour l'axe Y.

Remarques

Système de coordonnées 2D

Les transformations sont réalisées selon un système de coordonnées 2D X / Y. L'axe des X va de droite à gauche et l'axe des Y descend vers le bas, comme le montre l'image suivante:



Donc, un `translateY()` positif `translateY()` descend et un `translateX()` positif `translateX()` va bien.

Support du navigateur et préfixes

- IE prend en charge cette propriété depuis IE9 avec le préfixe `-ms-` . Les anciennes versions et Edge n'ont pas besoin du préfixe
- Firefox le supporte depuis la version 3.5 et nécessite le préfixe `-moz-` jusqu'à la version 15
- Chrome depuis la version 4 et jusqu'à la version 34 nécessite le préfixe `-webkit-`
- Safari a besoin du préfixe `-webkit-` jusqu'à la version 8
- Opera a besoin du préfixe `-o-` pour la version 11.5 et du préfixe `-webkit-` de la version 15 à 22
- Android a besoin du préfixe `-webkit-` de la version 2.1 à la version 4.4.4

Exemple de transformation préfixée:

```
-webkit-transform: rotate(45deg);
    -ms-transform: rotate(45deg);
        transform: rotate(45deg);
```

Examples

Tourner

HTML

```
<div class="rotate"></div>
```

CSS

```
.rotate {
    width: 100px;
    height: 100px;
    background: teal;
    transform: rotate(45deg);
}
```

Cet exemple fera pivoter le div de 45 degrés dans le sens des aiguilles d'une montre. Le centre de rotation se trouve au centre du div, à `50%` de la gauche et à `50%` du haut. Vous pouvez modifier le centre de rotation en définissant la propriété d' `transform-origin` .

```
transform-origin: 100% 50%;
```

L'exemple ci-dessus définira le centre de rotation au milieu de l'extrémité droite.

Échelle

HTML

```
<div class="scale"></div>
```

CSS

```
.scale {
    width: 100px;
    height: 100px;
    background: teal;
    transform: scale(0.5, 1.3);
}
```

Cet exemple mettra à l'échelle le div sur `100px * 0.5 = 50px` sur l'axe des abscisses et sur `100px * 1.3 = 130px` sur l'axe des `100px * 1.3 = 130px` .

Le centre de la transformation se situe au centre du div, à `50%` de la gauche et à `50%` du haut.

Traduire

HTML

```
<div class="translate"></div>
```

CSS

```
.translate {
    width: 100px;
    height: 100px;
    background: teal;
    transform: translate(200px, 50%);
}
```

Cet exemple déplace le div de 200px sur l'axe des x et de `100px * 50% = 50px` sur l'axe des y.

Vous pouvez également spécifier des traductions sur un seul axe.

Sur l'axe des X:

```
.translate {
    transform: translateX(200px);
}
```

Sur l'axe des Y:

```
.translate {
    transform: translateY(50%);
}
```

Fausser

HTML

```
<div class="skew"></div>
```

CSS

```
.skew {
    width: 100px;
    height: 100px;
    background: teal;
    transform: skew(20deg, -30deg);
}
```

Cet exemple incline le div de 20 degrés sur l'axe des X et de - 30 degrés sur l'axe des Y.
Le centre de la transformation se situe au centre du div, à `50%` de la gauche et à `50%` du haut.

Voir le résultat ici .

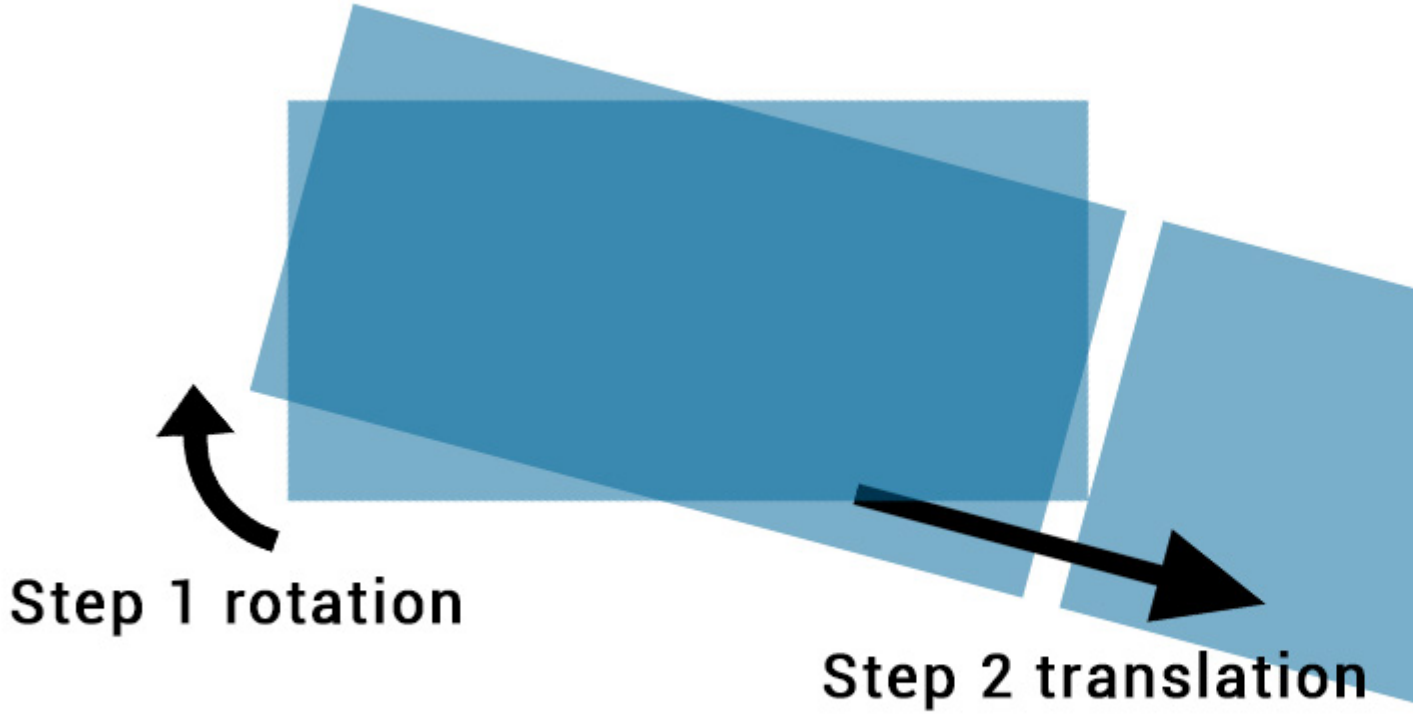
Transformations multiples

Plusieurs transformations peuvent être appliquées à un élément d'une propriété comme ceci:

```
transform: rotate(15deg) translateX(200px);
```

Cela fera pivoter l'élément de 15 degrés dans le sens des aiguilles d'une montre et le traduira ensuite à 200px vers la droite.

Dans les transformations chaînées, **le système de coordonnées se déplace avec l'élément** .
Cela signifie que la traduction ne sera pas horizontale, mais que sur un axe, pivotez de 15 degrés dans le sens des aiguilles d'une montre, comme indiqué dans l'image suivante:



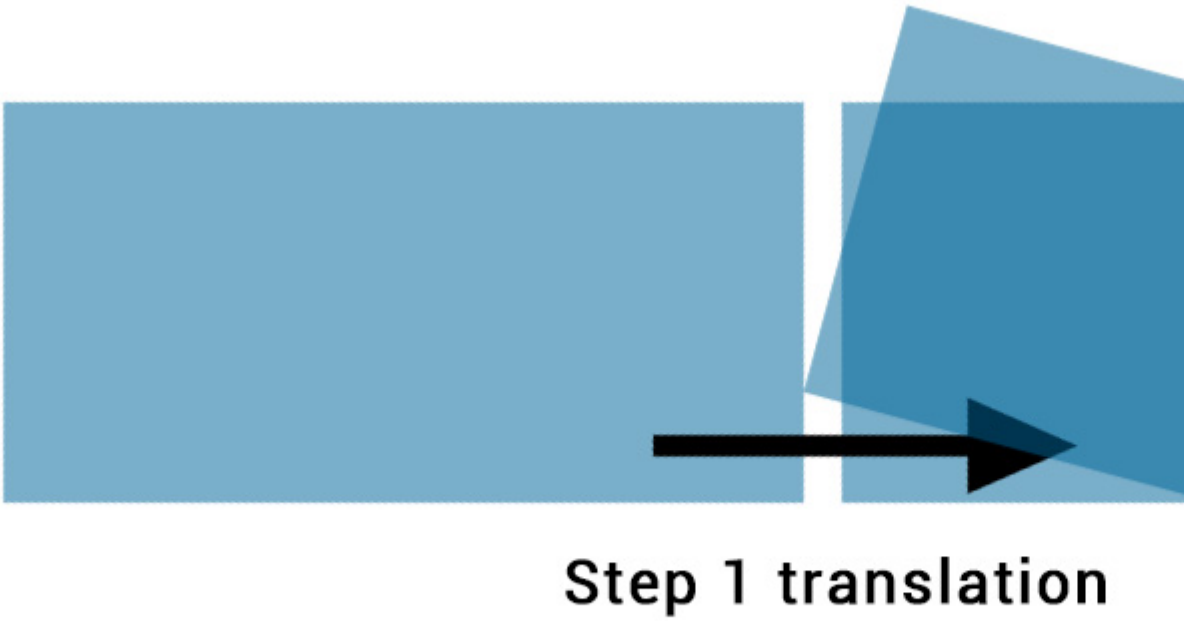
Changer l'ordre des transformations changera la sortie. Le premier exemple sera différent de

```
transform: translateX(200px) rotate(15deg);
```

```
<div class="transform"></div>
```

```
.transform {
  transform: rotate(15deg) translateX(200px);
}
```

Comme montré dans cette image:



Transformer l'origine

Les transformations sont effectuées par rapport à un point défini par la propriété `transform-origin` .

La propriété prend 2 valeurs: `transform-origin: XY;`

Dans l'exemple suivant, la première div (`.tl`) est `.tl` dans le coin supérieur gauche avec l' `transform-origin: 0 0;` et le second (`.tr`) est transformé autour de son coin supérieur droit avec l' `transform-origin: 100% 0` . La rotation est appliquée **en vol stationnaire** :

HTML:

```
<div class="transform originl"></div>
<div class="transform origin2"></div>
```

CSS:

```
.transform {
    display: inline-block;
    width: 200px;
    height: 100px;
    background: teal;
    transition: transform 1s;
}

.origin1 {
```

```
    transform-origin: 0 0;
}

.origin2 {
    transform-origin: 100% 0;
}

.transform:hover {
    transform: rotate(30deg);
}
```

La valeur par défaut de la propriété d'origine de transformation est `50% 50%` ce qui correspond au centre de l'élément.

Lire Transformations 2D en ligne: https://riptutorial.com/fr/css/topic/938/transformations-2d

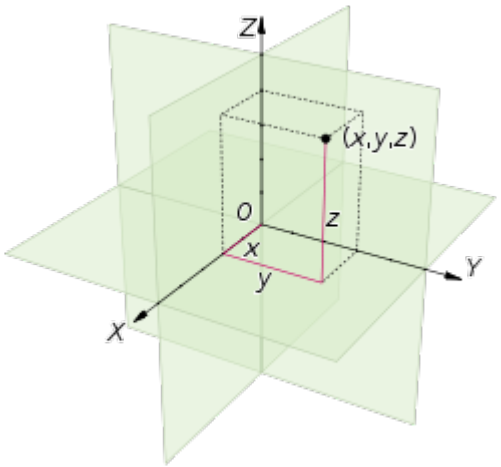
Chapitre 53: Transformations 3D

Remarques

Système de coordonnées

Les transformations 3D sont effectuées selon un système de vecteurs de coordonnées `(x, y, z)` dans l' espace euclidien .

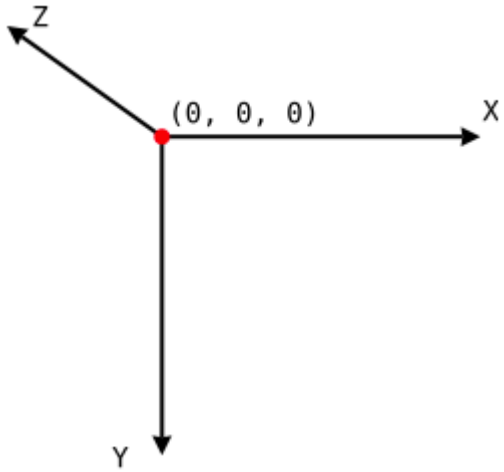
L'image suivante montre un exemple de coordonnées dans l'espace euclidien:



En CSS,

- L'axe des `x` représente l'horizontale (gauche et droite)
- L'axe des `y` représente la verticale (haut et bas)
- L'axe `z` représente la profondeur (avant et arrière / plus proche et plus loin)

L'image suivante montre comment ces coordonnées sont traduites en CSS:



Examples

Cube 3D

Les transformations 3D peuvent être utilisées pour créer de nombreuses formes 3D. Voici un exemple simple de cube CSS 3D:

HTML:

```
<div class="cube">
  <div class="cubeFace"></div>
  <div class="cubeFace face2"></div>
</div>
```

CSS:

```
body {
  perspective-origin: 50% 100%;
  perspective: 1500px;
  overflow: hidden;
}
.cube {
  position: relative;
  padding-bottom: 20%;
  transform-style: preserve-3d;
  transform-origin: 50% 100%;
  transform: rotateY(45deg) rotateX(0);
}
.cubeFace {
  position: absolute;
  top: 0;
  left: 40%;
  width: 20%;
  height: 100%;
  margin: 0 auto;
  transform-style: inherit;
  background: #C52329;
  box-shadow: inset 0 0 0 5px #333;
  transform-origin: 50% 50%;
  transform: rotateX(90deg);
  backface-visibility: hidden;
}
.face2 {
  transform-origin: 50% 50%;
  transform: rotatez(90deg) translateX(100%) rotateY(90deg);
}
.cubeFace:before, .cubeFace:after {
  content: '';
  position: absolute;
  width: 100%;
  height: 100%;
  transform-origin: 0 0;
  background: inherit;
  box-shadow: inherit;
  backface-visibility: inherit;
}
.cubeFace:before {
  top: 100%;
  left: 0;
  transform: rotateX(-90deg);
```

```
}
.cubeFace:after {
  top: 0;
  left: 100%;
  transform: rotateY(90deg);
}
```

Voir cet exemple
Un style supplémentaire est ajouté dans la démo et une transformation est appliquée lors du survol pour afficher les 6 faces du cube.

Devrait être noté que:

- 4 faces sont faites avec des pseudo-éléments
- les transformations chaînées sont appliquées

visibilité sur la face arrière

La propriété de `backface-visibility` concerne les transformations 3D.

Grâce aux transformations 3D et à la propriété de `backface-visibility` , vous pouvez faire pivoter un élément de sorte que le recto d'origine d'un élément ne soit plus en regard de l'écran.

Par exemple, cela ferait disparaître un élément de l'écran:

JSFIDDLE

```
<div class="flip">Loren ipsum</div>
<div class="flip back">Lorem ipsum</div>
```

```
.flip {
  -webkit-transform: rotateY(180deg);
  -moz-transform: rotateY(180deg);
  -ms-transform: rotateY(180deg);
  -webkit-backface-visibility: visible;
  -moz-backface-visibility:    visible;
  -ms-backface-visibility:     visible;
}

.flip.back {
  -webkit-backface-visibility: hidden;
  -moz-backface-visibility:    hidden;
  -ms-backface-visibility:     hidden;
}
```

Firefox 10+ et IE 10+ prennent en charge la `backface-visibility` sans préfixe. Opera, Chrome, Safari, iOS et Android nécessitent tous une `-webkit-backface-visibility` .

Il a 4 valeurs:

1. **visible** (par défaut) - l'élément sera toujours visible même lorsque vous ne faites pas face à l'écran.
2. **hidden** - l'élément n'est pas visible lorsqu'il n'est pas face à l'écran.

3. **inherit** - la propriété va obtenir sa valeur de l'élément parent
4. **initial** - définit la propriété à sa valeur par défaut, qui est visible

Compas pointeur ou forme d'aiguille à l'aide de transformations 3D

CSS

```
div.needle {
  margin: 100px;
  height: 150px;
  width: 150px;
  transform: rotateY(85deg) rotateZ(45deg);
  /* presentational */
  background-image: linear-gradient(to top left, #555 0%, #555 40%, #444 50%, #333 97%);
  box-shadow: inset 6px 6px 22px 8px #272727;
}
```

HTML

```
<div class='needle'></div>
```

Dans l'exemple ci-dessus, une forme de pointeur d'aiguille ou de boussole est créée à l'aide de transformations 3D. Généralement, lorsque nous appliquons la transformation de `rotate` sur un élément, la rotation ne se produit que sur l'axe des Z et au mieux, nous nous retrouverons avec des formes de diamant uniquement. Mais lorsqu'une transformation `rotateY` est ajoutée par dessus, l'élément est comprimé sur l'axe des Y et finit par ressembler à une aiguille. Plus la rotation de l'axe des Y est importante, plus l'élément est compressé.

La sortie de l'exemple ci-dessus serait une aiguille posée sur sa pointe. Pour créer une aiguille qui repose sur sa base, la rotation doit se faire sur l'axe des abscisses plutôt que sur l'axe des ordonnées. La valeur de la propriété `transform` devrait donc être quelque chose comme `rotateX(85deg) rotateZ(45deg);` .

Ce stylet utilise une approche similaire pour créer quelque chose qui ressemble au logo Safari ou à un cadran de la boussole.

Capture d'écran de l'élément sans transformation:

Capture d'écran de l'élément avec uniquement une transformation 2D:

Capture d'écran d'élément avec transformation 3D:

Effet de texte 3D avec ombre

HTML:

```
<div id="title">
  <h1 data-content="HOVER">HOVER</h1>
</div>
```

CSS:

```
*{margin:0;padding:0;}
html,body{height:100%;width:100%;overflow:hidden;background:#0099CC;}
#title{
  position:absolute;
  top:50%; left:50%;
  transform:translate(-50%,-50%);
  perspective-origin:50% 50%;
  perspective:300px;
}
h1{
  text-align:center;
  font-size:12vmin;
  font-family: 'Open Sans', sans-serif;
```

```
  color:rgba(0,0,0,0.8);
  line-height:1em;
  transform:rotateY(50deg);
  perspective:150px;
  perspective-origin:0% 50%;
}
h1:after{
  content:attr(data-content);
  position:absolute;
  left:0;top:0;
  transform-origin:50% 100%;
  transform:rotateX(-90deg);
  color:#0099CC;
}
#title:before{
  content:'';
  position:absolute;
  top:-150%; left:-25%;
  width:180%; height:328%;
  background:rgba(255,255,255,0.7);
  transform-origin: 0 100%;
  transform: translatez(-200px) rotate(40deg) skewX(35deg);
  border-radius:0 0 100% 0;
}
```

Voir l'exemple avec un effet de survol supplémentaire

Dans cet exemple, le texte est transformé pour donner l'impression qu'il entre à l'écran de l'utilisateur.

L'ombre est transformée en conséquence pour suivre le texte. Comme il est fait avec un pseudo-

élément et l'attribut `data` , il hérite des transformations de son parent (la balise H1).

La "lumière" blanche est faite avec un pseudo-élément sur l'élément `#title` . Il est incliné et utilise border-radius pour le coin arrondi.

Lire Transformations 3D en ligne: https://riptutorial.com/fr/css/topic/2446/transformations-3d

Chapitre 54: Transitions

Syntaxe

- transition: [propriété de transition] [durée de transition] [fonction de synchronisation de transition] [délai de transition];

Paramètres

Paramètre	Détails
propriété de transition	La propriété CSS spécifique dont la valeur doit être modifiée (ou) `all` , si toutes les propriétés de transition doivent être transférées.
durée de transition	La durée (ou la période) en secondes (`s`) ou millisecondes (`ms`) sur laquelle la transition doit avoir lieu.
fonction de transition-timing	Une fonction qui décrit comment les valeurs intermédiaires pendant la transition sont calculées. Les valeurs couramment utilisées sont la `ease` , la `ease-in` , la `ease-out` , la `ease-in-out` , la `linear` , le `cubic-bezier()` , les `steps()` . Vous trouverez plus d'informations sur les différentes fonctions de synchronisation dans les spécifications du W3C .
retard de transition	La quantité de temps qui doit s'être écoulée avant que la transition ne puisse commencer. Peut être spécifié en secondes (`s`) ou millisecondes (`ms`)

Remarques

Certains anciens navigateurs ne prennent en charge que les propriétés de `transition` préfixées par le fournisseur :

- `-webkit` : Chrome 25-, Safari 6-, Safari et Chrome pour iOS 6.1-, Android 4.3- Navigateur, Blackberry Browser 7-, UC Browser 9.9- pour Android.
- `-moz` : Firefox 15-.
- `-o` : Opera 11.5-, Opera Mobile 12-.

Exemple:

```
-webkit-transition: all 1s;
-moz-transition: all 1s;
-o-transition: all 1s;
transition: all 1s;
```

Examples

Sténographie de transition

CSS

```
div{
    width: 150px;
    height:150px;
    background-color: red;
    transition: background-color 1s;
}
div:hover{
    background-color: green;
}
```

HTML

```
<div></div>
```

Cet exemple changera la couleur de fond lorsque le div sera survolé, le changement de couleur d'arrière-plan durera 1 seconde.

Transition (longhand)

CSS

```
div {
  height: 100px;
  width: 100px;
  border: 1px solid;
  transition-property: height, width;
  transition-duration: 1s, 500ms;
  transition-timing-function: linear;
  transition-delay: 0s, 1s;
}
div:hover {
  height: 200px;
  width: 200px;
}
```

HTML

```
<div></div>
```

- **transition-property** : Spécifie les propriétés CSS de l'effet de transition. Dans ce cas, le div se développera horizontalement et verticalement lorsqu'il sera survolé.
- **transition-duration** : Spécifie la durée d' **exécution** d'une transition. Dans l'exemple ci-dessus, les transitions en hauteur et en largeur prendront respectivement 1 seconde et 500 millisecondes.

- **transition-timing-function** : Spécifie la courbe de vitesse de l'effet de transition. Une valeur *linéaire* indique que la transition aura la même vitesse du début à la fin.
- **transition-delay** : Spécifie la durée d'attente avant le démarrage de l'effet de transition. Dans ce cas, la hauteur commencera à faire la transition immédiatement, alors que la largeur attendra 1 seconde.

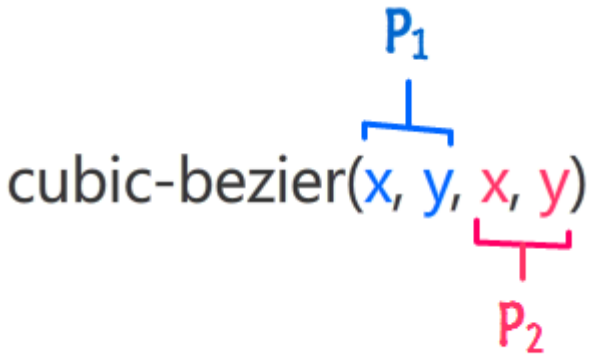
cubic-bezier

La fonction `cubic-bezier` est une fonction de synchronisation de transition qui est souvent utilisée pour des transitions personnalisées et fluides.

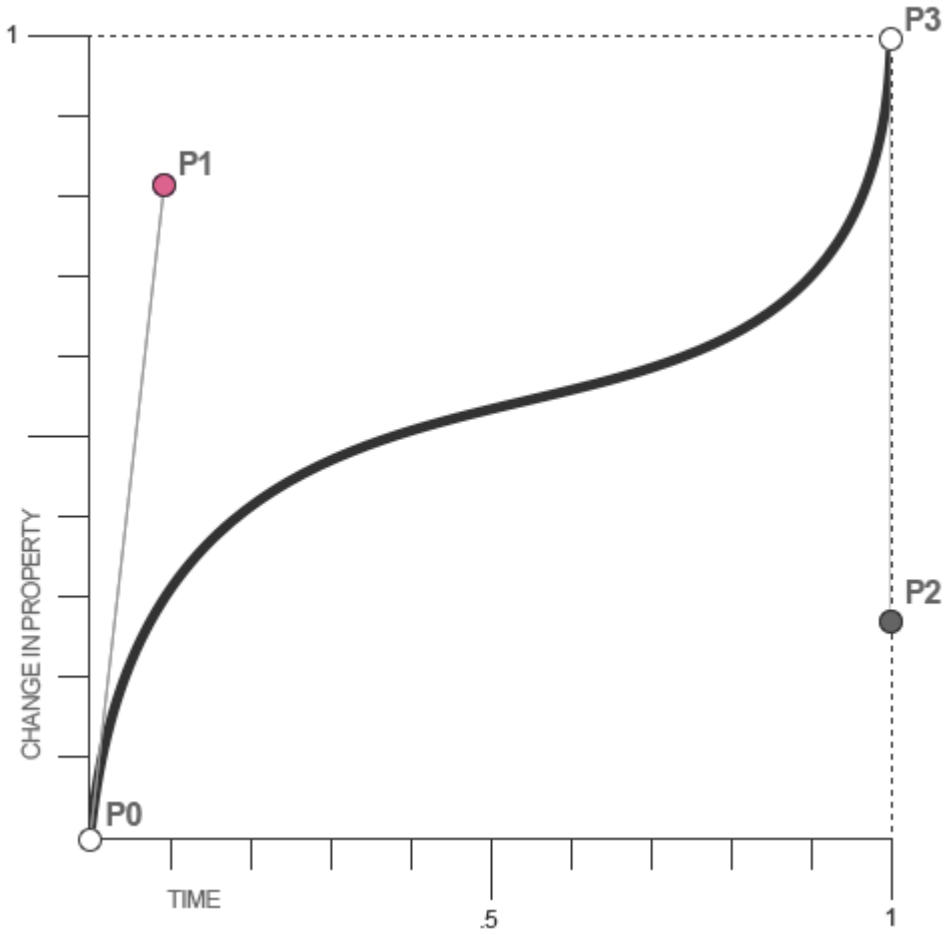
```
transition-timing-function: cubic-bezier(0.1, 0.7, 1.0, 0.1);
```

La fonction prend quatre paramètres:

```
cubic-bezier(P1_x, P1_y, P2_x, P2_y)
```



Ces paramètres seront mappés sur des points faisant partie d'une courbe de Bézier :



Pour les courbes CSS de Bézier, P0 et P3 sont toujours au même endroit. P0 est à (0,0) et P3 à (1,1), ce qui signifie que les paramètres passés à la fonction cubique-bezier ne peuvent être qu'entre 0 et 1.

Si vous passez des paramètres qui ne sont pas dans cet intervalle, la fonction passera par défaut à une transition `linear` .

Comme cubic-bezier est la transition la plus flexible en CSS, vous pouvez traduire toutes les autres fonctions de synchronisation de la transition en fonctions cubiques-bezier:

`linear` : `cubic-bezier(0,0,1,1)`

`ease-in` : `cubic-bezier(0.42, 0.0, 1.0, 1.0)`

`ease-out` : `cubic-bezier(0.0, 0.0, 0.58, 1.0)`

`ease-in-out` : `cubic-bezier(0.42, 0.0, 0.58, 1.0)`

Lire Transitions en ligne: https://riptutorial.com/fr/css/topic/751/transitions

Chapitre 55: Typographie

Syntaxe

- font: [*font-style*] *[font-variant] [font-weight] font-size [/ line-height] font-family* ;
- style de *police* : style de *police*
- variante de *police* : variante de *police*
- poids de la *police* : poids de la *police* ;
- taille de la *police* : taille de la *police* ;
- line-height: hauteur de la *ligne* ;
- font-family: *font-family* ;
- couleur: *couleur* ;
- quotes: *none* | *string* | *initial* | *inherit* ;
- font-stretch: *font-stretch* ;
- text-align: *text-align* ;
- text-indent: *length* | *initial* | *inherit* ;
- text-overflow: *clip* | *ellipsis* | *string* | *initial* | *inherit* ;
- text-transform: *none* | *majuscule* | *majuscule* | *minuscule* | *initial* |
- text-shadow: *h-ombre v-shadow blur-radius couleur* | *none* | *initial* | *hérite* ;
- font-size-adjust: *number* | *none* | *initial* | *inherit;*
- font-stretch: *ultra-condensé* | *extra-condensé* | *condensé* | *semi-condensé* | *normal* | *semi-étendu* | *développé* | *extra-étendu* | *ultra-étendu* | *initial* | *hérité;*
- traits d'union: *aucun* | *manuel* | *auto* ;
- tab-size: *nombre* | *length* | *initial* | *inherit* ;
- interligne: *normal* | *length* | *initial* | *inherit* ;
- espacement des mots: *normal* | *length* | *initial* | *inherit* ;

Paramètres

Paramètre	Détails
le style de police	`italics` ou `oblique`
variante de police	`normal` ou `small-caps`
poids de la police	`normal` , `bold` ou numérique de 100 à 900.
taille de police	La taille de la police donnée en `%` , `px` , `em` ou toute autre mesure CSS valide
hauteur de la ligne	La hauteur de ligne donnée en `%` , `px` , `em` ou toute autre mesure CSS valide

Paramètre	Détails
famille de polices	C'est pour définir le nom de la famille.
Couleur	Toute représentation de couleur CSS valide, comme le `red` , `#00FF00` , `hsl(240, 100%, 50%)` etc.
étirer la police	Utiliser ou non un visage confisqué ou étendu à partir de la police. Les valeurs valides sont `normal` , `ultra-condensed` , `extra-condensed` , `condensed` , `semi-condensed` , `semi-expanded` , `expanded` , `extra-expanded` ou `ultra-expanded`
aligner le texte	`start` , `end` , `left` , `right` , `center` , `justify` , `match-parent`
décoration de texte	`none` , `underline` , `overline` , `line-through` , `initial` , `inherit` ;

Remarques

- La propriété `text-shadow` n'est pas prise en charge par les versions d'Internet Explorer inférieures à 10.

Examples

Taille de police

HTML:

```
<div id="element-one">Hello I am some text.</div>
<div id="element-two">Hello I am some smaller text.</div>
```

CSS:

```
#element-one {
    font-size: 30px;
}

#element-two {
    font-size: 10px;
}
```

Le texte à l'intérieur de `#element-one` aura une taille de `30px` , tandis que le texte dans `#element-two` aura une taille de `10px` .

La sténographie de la police

Avec la syntaxe:

```
element {
    font: [font-style] [font-variant] [font-weight] [font-size/line-height] [font-family];
}
```

Vous pouvez avoir tous les styles liés aux polices dans une seule déclaration avec le raccourci de `font` . Utilisez simplement la propriété `font` et mettez vos valeurs dans le bon ordre.

Par exemple, pour que tous les éléments `p` soient en gras avec une taille de police de 20 pixels et que vous utilisiez Arial comme famille de polices, vous devez le coder comme suit:

```
p {
    font-weight: bold;
    font-size: 20px;
    font-family: Arial, sans-serif;
}
```

Cependant, avec le raccourci de police, il peut être condensé comme suit:

```
p {
    font: bold 20px Arial, sans-serif;
}
```

Remarque : étant donné que `font-style` , `font-variant` , `font-weight` et `line-height` sont facultatifs, ils sont tous trois ignorés dans cet exemple. Il est important de noter que l'utilisation du raccourci **réinitialise** les autres attributs non fournis. Un autre point important est que les deux attributs nécessaires au raccourci de police pour travailler sont `font-size` `font-family` et `font-family` . S'ils ne sont pas tous les deux inclus, le raccourci est ignoré.

Valeur initiale pour chacune des propriétés:

- `font-style: normal;`
- `font-variant: normal;`
- `font-weight: normal;`
- `font-stretch: normal;`
- `font-size: medium;`
- `line-height: normal;`
- `font-family` - dépend de l'agent utilisateur

Piles de polices

```
font-family: 'Segoe UI', Tahoma, sans-serif;
```

Le navigateur tentera d'appliquer la police "Segoe UI" aux caractères des éléments ciblés par la propriété ci-dessus. Si cette police n'est pas disponible ou si la police ne contient pas de glyphe pour le caractère requis, le navigateur utilisera Tahoma et, si nécessaire, toute police sans empattement sur l'ordinateur de l'utilisateur. Notez que tous les noms de police comportant plusieurs mots, tels que "Segoe UI", doivent contenir des guillemets simples ou doubles.

```
font-family: Consolas, 'Courier New', monospace;
```

Le navigateur tentera d'appliquer la police "Consolas" aux caractères des éléments ciblés par la propriété ci-dessus. Si cette police n'est pas disponible ou si la police ne contient pas de glyphe pour le caractère requis, le navigateur utilisera "Courier New" et, si nécessaire, toute police monospace sur l'ordinateur de l'utilisateur.

L'espacement des lettres

```
h2 {
  /* adds a 1px space horizontally between each letter;
     also known as tracking */
  letter-spacing: 1px;
}
```

La propriété d'espacement des lettres permet de spécifier l'espace entre les caractères d'un texte.

! l'espacement des lettres prend également en charge les valeurs négatives:

```
p {
  letter-spacing: -1px;
}
```

Ressources: https://developer.mozilla.org/en-US/docs/Web/CSS/letter-spacing

Transformation de texte

La propriété `text-transform` vous permet de modifier la capitalisation du texte. Les valeurs valides sont les suivantes: `uppercase` , `capitalize` , `lowercase` , `initial` , `inherit` et `none`

CSS:

```
.example1 {
    text-transform: uppercase;
}
.example2 {
    text-transform: capitalize;
}
.example3 {
    text-transform: lowercase;
}
```

HTML

```
<p class="example1">
    all letters in uppercase  <!-- "ALL LETTERS IN UPPERCASE" -->
</p>
<p class="example2">
    all letters in capitalize <!-- "All Letters In Capitalize (Sentence Case)" -->
</p>
<p class="example3">
    all letters in lowercase  <!-- "all letters in lowercase" -->
</p>
```

Retrait du texte

```
p {
    text-indent: 50px;
}
```

La propriété `text-indent` spécifie la quantité de texte d'espace horizontal à déplacer avant le début de la première ligne du contenu textuel d'un élément.

Ressources:

- En retrait de la première ligne de texte d'un paragraphe?
- https://www.w3.org/TR/CSS21/text.html#propdef-text-indent
- https://developer.mozilla.org/en-US/docs/Web/CSS/text-indent

Décoration de texte

La propriété `text-decoration` permet de définir ou de supprimer des décorations du texte.

```
h1 { text-decoration: none; }
h2 { text-decoration: overline; }
h3 { text-decoration: line-through; }
h4 { text-decoration: underline; }
```

la décoration de texte peut être utilisée en combinaison avec le style de décoration de texte et la couleur de décoration de texte:

```
.title { text-decoration: underline dotted blue; }
```

Ceci est une version abrégée de

```
.title {
  text-decoration-style: dotted;
  text-decoration-line: underline;
  text-decoration-color: blue;
}
```

Il convient de noter que les propriétés suivantes sont uniquement prises en charge dans Firefox

- texte-décoration-couleur
- ligne de décoration de texte
- style de décoration de texte
- texte-décoration-sauter

Dépassement de texte

La propriété `text-overflow` traite de la manière dont le contenu débordé doit être signalé aux utilisateurs. Dans cet exemple, les `ellipsis` représentent le texte tronqué.

```
.text {
  overflow: hidden;
  text-overflow: ellipsis;
}
```

Malheureusement, `text-overflow: ellipsis` ne fonctionnent que sur une seule ligne de texte. Il n'y a aucun moyen de prendre en charge les points de suspension sur la dernière ligne des CSS standard, mais cela peut être réalisé avec une implémentation non standard de webbox uniquement pour les flexbox.

```
.giveMeEllipsis {
   overflow: hidden;
   text-overflow: ellipsis;
   display: -webkit-box;
   -webkit-box-orient: vertical;
   -webkit-line-clamp: N; /* number of lines to show */
   line-height: X;        /* fallback */
   max-height: X*N;       /* fallback */
}
```

Exemple (ouvert dans Chrome ou Safari):

http://jsfiddle.net/csYjC/1131/

Ressources:

https://www.w3.org/TR/2012/WD-css3-ui-20120117/#text-overflow0

Espacement des mots

La propriété d'espacement des mots spécifie le comportement d'espacement entre les balises et les mots.

Valeurs possibles

- une *longueur* positive ou négative (en utilisant `em px vh cm` etc.) ou un *pourcentage* (en utilisant `%`)
- le mot clé `normal` utilise l'espacement des mots par défaut de la police
- le mot clé `inherit` prend la valeur de l'élément parent

CSS

```
.normal    { word-spacing: normal; }
.narrow    { word-spacing: -3px; }
.extensive { word-spacing: 10px; }
```

HTML

```
<p>
  <span class="normal">This is an example, showing the effect of "word-spacing".</span><br>
  <span class="narrow">This is an example, showing the effect of "word-spacing".</span><br>
  <span class="extensive">This is an example, showing the effect of "word-spacing".</span><br>
```

```
</p>
```

Démo en ligne

Essayez vous-même

Lectures complémentaires:

- espacement des mots - MDN
- espacement des mots - w3.org

Direction du texte

```
div {
    direction: ltr; /* Default, text read read from left-to-right */
}
.ex {
    direction: rtl; /* text read from right-to-left */
}
.horizontal-tb {
    writing-mode: horizontal-tb; /* Default, text read from left-to-right and top-to-bottom.
*/
}
.vertical-rtl {
    writing-mode: vertical-rl; /* text read from right-to-left and top-to-bottom */
}
.vertical-ltr {
    writing-mode: vertical-rl; /* text read from left-to-right and top to bottom */
}
```

La propriété `direction` est utilisée pour modifier la direction du texte horizontal d'un élément.

Syntaxe: `direction: ltr | rtl | initial | inherit;`

La propriété du `writing-mode` modifie l'alignement du texte afin qu'il puisse être lu de haut en bas ou de gauche à droite, selon la langue.

Syntaxe: `direction: horizontal-tb | vertical-rl | vertical-lr;`

Variante de police

Les attributs:

Ordinaire

Attribut par défaut des polices.

en minuscule

Définit chaque lettre en majuscule, **mais** rend les lettres minuscules (à partir du texte original) plus petites que les lettres en majuscules.

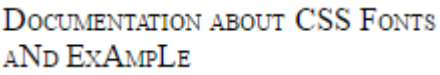
CSS:

```
.smallcaps{
   font-variant: small-caps;
}
```

HTML:

```
<p class="smallcaps">
    Documentation about CSS Fonts
    <br>
    aNd ExAmpLe
</p>
```

OUPUT:



Remarque: La propriété font-variant est un raccourci pour les propriétés: font-variant-caps, font-variant-numeric, font-variant-alternates, font-variant-ligatures et font-variant-east-asian.

Citations

La propriété `quotes` permet de personnaliser les guillemets d'ouverture et de fermeture de la `<q>` .

```
q {
    quotes: "«" "»";
}
```

Ombre de texte

Pour ajouter des ombres au texte, utilisez la propriété `text-shadow` . La syntaxe est la suivante:

```
text-shadow: horizontal-offset vertical-offset blur color;
```

Ombre sans rayon de flou

```
h1 {
  text-shadow: 2px 2px #0000FF;
}
```

Cela crée un effet d'ombre bleue autour d'un titre

Ombre avec rayon de flou

Pour ajouter un effet de flou, ajoutez une option d'argument de `blur radius`

```
h1 {
```

```
  text-shadow: 2px 2px 10px #0000FF;
}
```

Plusieurs ombres

Pour donner à un élément plusieurs ombres, séparez-les par des virgules

```
h1 {
    text-shadow: 0 0 3px #FF0000, 0 0 5px #0000FF;
}
```

Lire Typographie en ligne: https://riptutorial.com/fr/css/topic/427/typographie

Chapitre 56: Unités de longueur

Introduction

Une mesure de distance CSS est un nombre immédiatement suivi d'une unité de longueur (px, em, pc, in,…)

CSS prend en charge un certain nombre d'unités de mesure de longueur. Ils sont absolus ou relatifs.

Syntaxe

- **unité de** *valeur*
- 1em

Paramètres

Unité	La description
%	Définir les tailles en termes d'objets parents ou d'objet en cours dépendant de la propriété
em	Relatif à la taille de la police de l'élément (2em signifie 2 fois la taille de la police en cours)
rem	Relatif à la taille de la police de l'élément racine
vw	Relatif à 1% de la largeur de la fenêtre d'affichage *
vh	Relatif à 1% de la hauteur de la fenêtre d'affichage *
vmin	Par rapport à 1% de la dimension * de la fenêtre d'affichage
vmax	Par rapport à 1% de la plus grande dimension de la fenêtre d'affichage *
cm	centimètres
mm	millimètres
dans	pouces (1in = 96px = 2.54cm)
px	pixels (1px = 1 / 96ème de 1in)
pt	points (1pt = 1/72 de 1in)
pc	picas (1pc = 12 pt)

Unité	La description
s	secondes (utilisé pour les animations et les transitions)
Mme	millisecondes (utilisé pour les animations et les transitions)
ex	Relatif à la hauteur x de la police actuelle
ch	Basé sur la largeur du caractère zéro (0)
fr	unité fractionnaire (utilisée pour la disposition de grille CSS)

Remarques

- Un espace ne peut pas apparaître entre le nombre et l'unité. Cependant, si la valeur est 0, l'unité peut être omise.
- Pour certaines propriétés CSS, les longueurs négatives sont autorisées.

Examples

Taille de la police avec rem

CSS3 introduit quelques nouvelles unités, y compris l'unité `rem` , qui signifie "root em". Regardons comment fonctionne `rem` .

Tout d'abord, regardons les différences entre `em` et `rem` .

- **em** : par rapport à la taille de la police du parent. Cela provoque le problème de la composition
- **rem** : par rapport à la taille de la police de l'élément root ou `<html>` . Cela signifie qu'il est possible de déclarer une taille de police unique pour l'élément html et de définir toutes les unités `rem` comme pourcentage.

Le principal problème avec l'utilisation de `rem` pour le dimensionnement des polices est que les valeurs sont quelque peu difficiles à utiliser. Voici un exemple de taille de police commune exprimée en unités `rem` , en supposant que la taille de base est 16px:

- 10px = 0.625rem
- 12px = 0.75rem
- 14px = 0.875rem
- 16px = 1rem (base)
- 18px = 1.125rem
- 20px = 1.25rem
- 24px = 1.5rem
- 30px = 1.875rem
- 32px = 2rem

CODE:

3

```
html {
  font-size: 16px;
}

h1 {
  font-size: 2rem;          /* 32px */
}

p {
  font-size: 1rem;          /* 16px */
}

li {
  font-size: 1.5em;         /* 24px */
}
```

Création d'éléments évolutifs à l'aide de rems et ems

3

Vous pouvez utiliser `rem` défini par la `font-size` de la `font-size` de votre balise `html` pour personnaliser les éléments en définissant leur `font-size` à une valeur `rem` et en utilisant `em` dans l'élément pour créer des éléments adaptés à votre `font-size` globale.

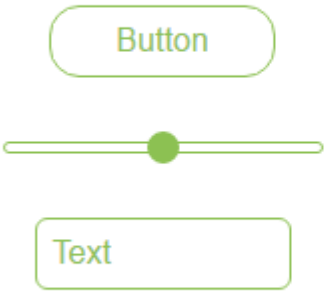
HTML:

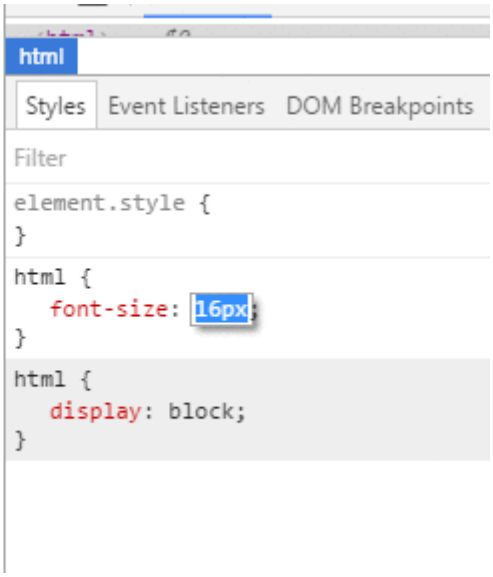
```
<input type="button" value="Button">
<input type="range">
<input type="text" value="Text">
```

CSS pertinent:

```
html {
  font-size: 16px;
}

input[type="button"] {
  font-size: 1rem;
  padding: 0.5em 2em;
}

input[type="range"] {
  font-size: 1rem;
  width: 10em;
}

input[type=text] {
  font-size: 1rem;
  padding: 0.5em;
}
```

Résultat possible:





vh et vw

CSS3 a introduit deux unités pour représenter la taille.

- `vh` , qui signifie que la `viewport height` d' `viewport height` est relative à 1% de la hauteur de la fenêtre d'affichage
- `vw` , qui signifie que la `viewport width` est relative à 1% de la largeur de la fenêtre d'affichage

3

```
div {
  width: 20vw;
  height: 20vh;
}
```

Au-dessus, la taille de la div occupe 20% de la largeur et de la hauteur de la fenêtre

vmin et vmax

- **vmin** : par rapport à 1% de la plus petite dimension de la fenêtre d' **affichage**
- **vmax** : par rapport à 1% de la plus grande dimension de la fenêtre d' **affichage**

En d'autres termes, 1 `vmin` est égal au plus petit de *1 vh* et *1 vw*

1 `vmax` est égal au plus grand de *1 vh* et *1 vw*

Remarque : `vmax` n'est pas pris en charge dans:

- toute version d'Internet Explorer
- Safari avant la version 6.1

en utilisant le pourcentage%

Une des unités utiles lors de la création d'une application réactive.

Sa taille dépend de son conteneur parent.

Équation:

(Largeur du conteneur parent) * (Pourcentage (%)) = Sortie

Par exemple:

Le parent a **une** largeur de **100px** alors que l' *enfant* a **50%** .

En sortie , la largeur de l' *enfant* sera la moitié (50%) de celle du *parent* , soit **50 pixels** .

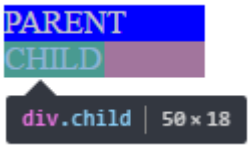
HTML

```
<div class="parent">
   PARENT
   <div class="child">
     CHILD
   </div>
</div>
```

CSS

```
<style>

*{
  color: #CCC;
}

.parent{
  background-color: blue;
  width: 100px;
}

.child{
  background-color: green;
  width: 50%;
}

</style>
```

SORTIE



Lire Unités de longueur en ligne: https://riptutorial.com/fr/css/topic/864/unites-de-longueur

Crédits

S. No	Chapitres	Contributeurs
1	Démarrer avec CSS	adamboro, animuson, Ashwin Ramaswami, awe, Boysenb3rry, Chris, Community, csx.cc, darrylyeo, FelipeAls, Gabriel R., Garconis, Gerardas, GoatsWearHats, G-Wiz, Harish Gyanani, Heri Hehe Setiawan, J Atkin, Jmh2013, joe_young, Jose Gomez , Just a student, Lambda Ninja, Marjorie Pickard, Nathan Arthur, patelarpan, RamenChef, Rocket Risa, Saroj Sasmal, ScientiaEtVeritas, selvassn, Sverri M. Olsen, Teo Dragovic, Todd, TylerH, Vivek Ghaisas, Xinyang Li, ZaneDickens
2	Ajustement d'objet et placement	4444, Miles, RamenChef
3	Arrière-plans	4444, Ahmad Alfy, animuson, Asim K T, Ben Rhys-Lewis, Boris, CalvT, cdm, Charlie H, CocoaBean, Dan Devine, Dan Eastwell, Daniel G. Blázquez, Daniel Stradowski, Darthstroke, designcise, Devid Farinelli, dippas, fcalderan, FelipeAls, Goose, Horst Jahns, Hynes, Jack, Jacob Gray, James Taylor, John Slegers, Jon Chan, Jonathan Zúñiga, Kevin Montrose, Louis St-Amour, Madalina Taina, Maximillian Laumeister, Michael Moriarty, Mr. Alien, mtb, Nate, Nathan Arthur, Nhan, Persijn, Praveen Kumar, RamenChef, Richard Hamilton, ScientiaEtVeritas, Sergey Denisov, Shaggy, Sourav Ghosh, Stewartside, Stratboy, think123, Timothy, Trevor Clarke, TylerH, Zac, Zeta, Zze
4	boîte ombre	Hristo, Madalina Taina, RamenChef
5	Cascade et spécificité	amflare, Arjan Einbu, brandaemon, DarkAjax, dippas, dmkerr, geeksal, Grant Palin, G-Wiz, James Donnelly, jehna1, John Slegers, kingcobra1986, Matas Vaitkevicius, Nathan Arthur, Nhan, Ortomala Lokni, RamenChef, ScientiaEtVeritas, Sunnyok , Ze Rubeus
6	Centrage	abaracedo, Alex Morales, Alohci, andreas, animuson, apaul, Christiaan Maks, Daniel Käfer, Devid Farinelli, Diego V, dippas, Eliran Malka, Emanuele Parisio, Euan Williams, F. Müller, Farzad YZ, Felix A J, geek1011, insertusernamehere, JedaiCoder, jehna1, JHS, John Slegers, Jonathan Argentiero, Kilian Stinson, Kyle Ratliff, Lambda Ninja, Lucia Bentivoglio, Luke Taylor, Madalina Taina, maho, Maxouhell, Michael_B, Mifeet, Mod Proxy, Mohammad Usman, Nathan Arthur, Nhan, o.v., Ortomala Lokni, paaacman, Paul Kozlovitch, Praveen

		Kumar, Sandeep Tuniki, ScientiaEtVeritas, Sergej, Siavas, smonff, Someone, Stewartside, Sunnyok, Taylor, TylerH, web-tiki, Ze Rubeus, zeel
7	Centrage vertical	animuson, AVAVT, bocanegra, Chiller, Chris, jaredsk, leo_ap, mmativ, patelarpan, Phil, Praveen Kumar, RamenChef
8	Colonnes	Brett DeWoody, Madalina Taina, RamenChef
9	commentaires	animuson, bdkopen, coderfin, Madalina Taina, Nick
10	Compteurs	Harry, RamenChef
11	Contexte d'empilement	Nemanja Trifunovic, RamenChef
12	Contextes de mise en forme de bloc	Madalina Taina, Milan Laslop, RamenChef
13	Contrôle de disposition	Arjun Iv, Chathuranga Jayanath, Chris, Jmh2013, Kevin Katzke, Kurtis Beavers, Madalina Taina, mnoronha, Niek Brouwer, RamenChef, Sander Koedood, ScientiaEtVeritas, SeinopSys
14	Couleurs	andreas, animuson, Arjan Einbu, Brett DeWoody, Community, cuervoo, darrylyeo, designcise, H. Pauwelyn, Jasmin Solanki, John Slegers, Kuhan, Marc, Michael Moriarty, Miro, Nathan Arthur, niyasc, RamenChef, Richard Hamilton, ScientiaEtVeritas, SeinopSys, Stewartside, user007, Wolfgang, X-27
15	Couper et Masquer	Andre Lopes, Harry, RamenChef, ScientiaEtVeritas, web-tiki
16	Débordement	Andrew, bdkopen, jgh, Madalina Taina, Miles, Qaz, RamenChef, ScientiaEtVeritas, Ted Goas, Zac
17	Des animations	Aeolingamenfel, apaul, Dex Star, Jasmin Solanki, Nathan Arthur, RamenChef, Richard Hamilton, TylerH
18	Des flotteurs	demonofthemist, FelipeAls, Madalina Taina, Nathan Arthur, RamenChef, vishak
19	Disposition de la boîte flexible (Flexbox)	Ahmad Alfy, Asim K T, FelipeAls, fzzylogic, James Donnelly, Jef, Lambda Ninja, Marc-Antoine Leclerc, Nathan Arthur, Nhan, Ori Marash, RamenChef, Randy, Squazz, takeradi, Ted Goas, Timothy Morris, TylerH
20	Disposition en ligne intégrée	Marten Koetsier, RamenChef
21	Formes à un seul élément	andreas, animuson, Brett DeWoody, Chiller, J F, Nick, RamenChef, ScientiaEtVeritas, TylerH

22	Formes pour flotteurs	animuson, Harry, RamenChef, ScientiaEtVeritas
23	Fragmentation	animuson, dodopok, Madalina Taina, Milan Laslop, RamenChef
24	Frontière	andreas, Cassidy Williams, doctorsherlock, FelipeAls, Gnietschow, Harry, jaredsk, Madalina Taina, Nobal Mohan, RamenChef, ScientiaEtVeritas, Trevor Clarke
25	Grandes lignes	Arif, Casey, Chathuranga Jayanath, Daniel Nugent, FelipeAls, Madalina Taina, RamenChef, ScientiaEtVeritas
26	Héritage	Chris, mnoronha, RamenChef
27	Internet Explorer Hacks	Aeolingamenfel, animuson, Elizaveta Revyakina, feeela, John Slegers, LiLacTac, Praveen Kumar, RamenChef, ScientiaEtVeritas, Timothy Miller, TylerH
28	la grille	Chris Spittles, FelipeAls, Jonathan Zúñiga, Mike McCaughan, Nhan, Praveen Kumar, RamenChef, Sebastian Zartner
29	Le modèle de boîte	Arjan Einbu, Ben Rhys-Lewis, Benolot, BiscuitBaker, FelipeAls, James Donnelly, Lambda Ninja, Nathan Arthur, Ortomala Lokni, RamenChef, ScientiaEtVeritas, Sergej, V-Kopio
30	Les fonctions	animuson, Brett DeWoody, cone56, dodopok, H. Pauwelyn, haim770, jaredsk, khawarPK, Kobi, RamenChef, SeinopSys, TheGenie OfTruth, TylerH
31	Les marges	Arjan Einbu, cdm, Chris Spittles, Community, J F, Madalina Taina, Mr_Green, Nathan Arthur, RamenChef, rejnev, Sun Qingyao, Sunnyok, Tot Zam, Trevor Clarke
32	les tables	Casper Spruit, Chris, FelipeAls, JHS, Madalina Taina
33	Modèle d'objet CSS (CSSOM)	Alohci, animuson, feeela, Paul Sweatte, RamenChef, rishabh dev
34	Modèles de conception CSS	John Slegers
35	Normaliser les styles de navigateur	andre mcgruder, Confused One, Grant Palin, MMachinegun, mnoronha, RamenChef, SeinopSys
36	Opacité	Andrew, animuson, Hillel Tech, Madalina Taina, RamenChef
37	Performance	mnoronha, RamenChef, TrungDQ
38	Plusieurs colonnes	bipon, Sebastian Zartner

39	Positionnement	Abhishek Singh, Alohci, animuson, Blunderfest, CalvT, Cassidy Williams, Chetan Joshi, GingerPlusPlus, Jacob Gray, Lambda Ninja, Madalina Taina, Matthew Beckman, Mattia Astorino, Rahul Nanwani, RamenChef, Sourav Ghosh, Stephen Leppik, Theodore K., TylerH
40	Propriété de filtre	Jeffery Tang, Nathan, RamenChef
41	Propriétés personnalisées (variables)	animuson, Brett DeWoody, Community, Daniel Käfer, Muthu Kumaran, Obsidian, RamenChef, RedRiderX, TylerH
42	Pseudo-éléments	4dgaurav, ankit, Chris, Community, dippas, dmnsgn, geek1011, geeksal, Gofilord, kevin vd, Marcatectura, Milche Patern, Nathan, Pat, Praveen Kumar, RamenChef, ScientiaEtVeritas, Shaggy, Stewartside, Sunnyok
43	Rembourrage	Andy G, CalvT, Felix A J, Madalina Taina, Mehdi Dehghani, Nathan, Paul Sweatte, pixelbandito, RamenChef, StefanBob, Will DiFruscio
44	Requêtes médias	amflare, Chathuranga Jayanath, darrylyeo, Demeter Dimitri, dodopok, James Donnelly, Jmh2013, joe_young, joejoe31b, John Slegers, Matas Vaitkevicius, Mattia Astorino, Maximillian Laumeister, Nathan Arthur, Praveen Kumar, RamenChef, ScientiaEtVeritas, srikarg, Teo Dragovic, Viktor
45	Requêtes sur les fonctionnalités	Andrew Myers, RamenChef
46	Sélecteurs	75th Trombone, A.J, Aaron, abaracedo, Ahmad Alfy, Alex Filatov, amflare, Anil, animuson, Araknid, Arjan Einbu, Ashwin Ramaswami, BoltClock, Cerbrus, Charlie H, Chris, Chris Nager, Clinton Yeboah, Community, CPHPython, darrylyeo, Dave Everitt, David Fullerton, Demeter Dimitri, designcise, Devid Farinelli, Devon Bernard, Dinidu, dippas, Erenor Paz, Felix Edelmann, Felix Schütz, flyingfisch, Forty, fracz, Frits, gandreadis, geeksal, George Bailey, George Grigorita, H. Pauwelyn, HansCz, henry, Hugo Buff, Hynes, J Atkin, J F, Jacob Gray, James Donnelly, James Taylor, Jasha, jehna1, Jmh2013, joejoe31b, Joël Bonet Rodríguez, John Slegers, Kurtis Beavers, Madalina Taina, Marc, Mark Perera, Matas Vaitkevicius, Matsemann, Michael_B, Milan Laslop, Naeem Shaikh, Nathan Arthur, Nick, Ortomala Lokni, Persijn, Praveen Kumar, RamenChef, rdans, Richard Hamilton, Rion Williams, Robert Koritnik, RockPaperLizard, RoToRa, Sbats, ScientiaEtVeritas, Shaggy, Siavas, Stewartside, sudo bangbang , Sumner Evans, Sunnyok, ThatWeirdo, theB, Thomas Gerot,

		TylerH, xpy, Yury Fedorov, Zac, Zaffy, Zaz, Ze Rubeus, Zze
47	Sprites d'image CSS	Elegant.Scripting, Jmh2013, RamenChef, Ted Goas, Ulrich Schwarz
48	Structure et mise en forme d'une règle CSS	Alon Eitan, darrylyeo, Marjorie Pickard
49	Style de curseur	cone56, Madalina Taina, ScientiaEtVeritas, Squazz
50	Styles de liste	animuson, Madalina Taina, Marten Koetsier, RamenChef, Ted Goas
51	Support du navigateur et préfixes	Andrew, animuson, Braiam, Nhan, Obsidian, RamenChef, ScientiaEtVeritas, Shaggy, TylerH
52	Transformations 2D	Charlie H, Christiaan Maks, Harry, Hors Sujet, John Slegers, Luke Taylor, Madalina Taina, mnoronha, PaMaDo, Praveen Kumar, RamenChef, web-tiki, zer00ne
53	Transformations 3D	Harry, Luka Kerr, Madalina Taina, mnoronha, Mr. Meeseeks, Nhan, RamenChef, web-tiki
54	Transitions	Christiaan Maks, dippas, Harry, Praveen Kumar, RamenChef, Richard Hamilton, ScientiaEtVeritas, Sergey Denisov, web-tiki
55	Typographie	Alex Morales, Alohci, andreas, Arjan Einbu, ChaoticTwist, Evgeny, Felix A J, Goulven, Hynes, insertusernamehere, James Donnelly, joe_young, Jon Chan, Madalina Taina, Michael Moriarty, Nathan Arthur, Nhan, Praveen Kumar, RamenChef, Richard Hamilton, rmondesilva, Ryan, Sourav Ghosh, Suhaib Janjua, Ted Goas, Toby, ToniB, Trevor Clarke, user2622348, Vlusion, Volker E.
56	Unités de longueur	4dgaurav, A B, animuson, Epodax, geeksal, J F, Marc, Milche Patern, Ortomala Lokni, RamenChef, Richard Hamilton, rmondesilva, Robert Koritnik, Robotnicka, ScientiaEtVeritas, StefanBob, Stewartside, Thomas Altmann, Toby, user2622348, vladdobra, Zakaria Acharki, zer00ne