Les fondamentaux du web (HTML, CSS, PHP)

Partie 2 - Le CSS

Valentin RIBEZZI

Sommaire

01

Les fondamentaux du web (HTML, CSS, PHP)

Partie 2 - Le CSS

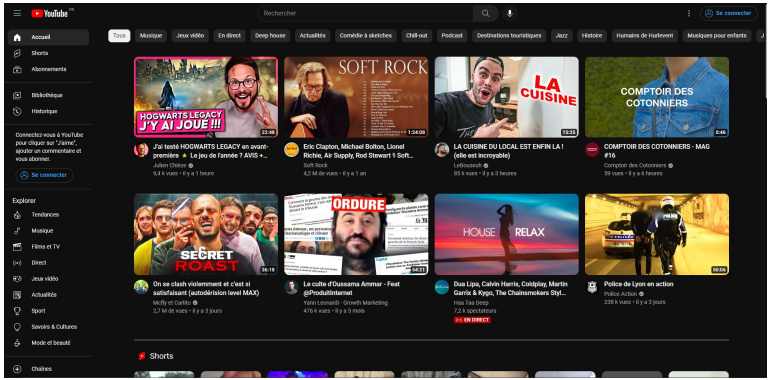
Présentation et objectifs du CSS

Présentation et objectifs du CSS

Les CSS (Cascading Style Sheets en anglais, ou « feuilles de style en cascade ») sont les fichiers utilisés pour mettre en forme une page web.

Concrètement, ils vont servir à mettre en forme le contenu que nous créons en HTML.

Exemple :



YOUTUBE avec du CSS

YOUTUBE sans du CSS

Présentation et objectifs du CSS

Où dois/peut-on écrire du CSS ?

- dans une balise <style> dans le fichier HTML (il est recommandé de les placer entre les balises <head>)

- Dans un fichier .css spécifique.
  Si cette option est choisie, il est indispensable d'importer le fichier CSS dans notre fichier HTML afin que celui-ci fonctionne.
  Pour cela, il faut utiliser la balise <link> :

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

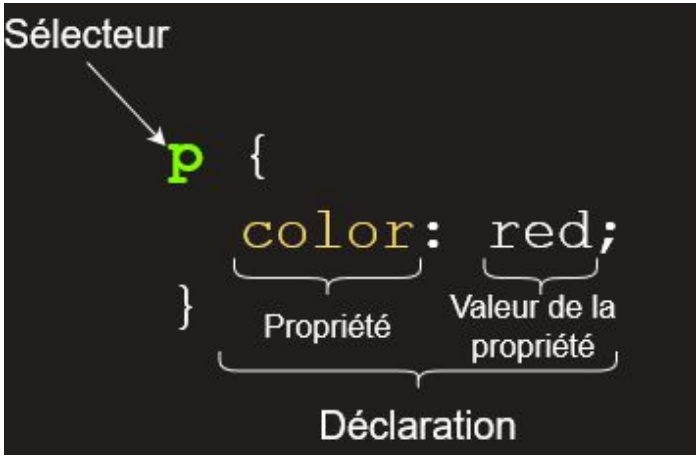
  (Attention, la balise <link> est une balise auto-fermante)

- Directement à l'intérieur d'une balise HTML à l'aide de l'attribut style
  ```
  <p style ="text-align : center; color: red;">...</p>
  ```

Présentation et objectifs du CSS

Comment fonctionne le CSS ?

Dans le principe, rien de très complexe :



Source : https://developer.mozilla.org/

02

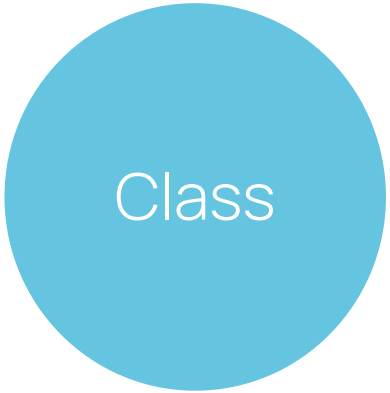
Les fondamentaux du web (HTML, CSS, PHP)

Partie 2 - Le CSS

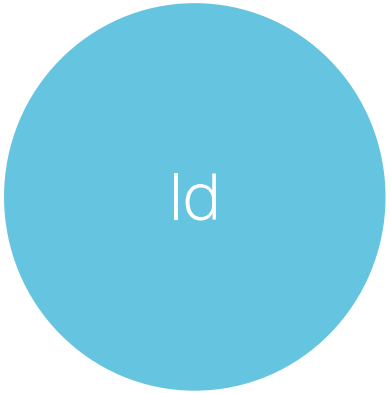
Les sélecteurs

Les sélecteurs

Il existe deux types d'attributs HTML que vous pourrez appeler en CSS :



Attribut servant à définir un ensemble de styles pour un ensemble de contenus.



Attribut unique servant à définir un ensemble de styles pour un contenu en particulier.

Rappel : mon paragraphe -> class sera l'attribut de notre balise

Les sélecteurs

```
<div id="white">
id = identificateur (unique)
  <span class="underline">Bonjour</span>
  class = class (universel /global)
  <p class="uppercase">test</p>
  class = class (universel /global)
</div>

<div id="black">
id = identificateur (unique)
  <span class="underline">Bonjour</span>
  class = class (universel /global)
  <p class="uppercase">test</p>
  class = class (universel /global)
</div>
```

```
#white {
  color: white;
}

#black {
  color: black;
}

.underline {
  text-decoration: underline:
}

.uppercase {
  text-transform: uppercase;
}
```

Les sélecteurs

Les sélecteurs simples

Les sélecteurs de type

HTML
```
<input>
```

CSS
```
input {...}
```

Les sélecteurs de classe (universel / global)

HTML
```
<input class="padding">
```

CSS
```
.padding {...}
```

Les sélecteurs d'identifiant (unique)

HTML
```
<input id="padding">
```

CSS
```
#padding {...}
```

Les sélecteurs

Les sélecteurs simples

Le sélecteur universel

HTML

```
.....
```

CSS

```
* {...}
```

Les sélecteurs d'attribut

HTML

```
<input class="padding">
```

CSS

```
[class="padding"] {...}
```

(Permet de cibler un élément qui possède la classe padding)

HTML

```
<input class="padding1">
```

CSS

```
[class$="1"] {...}
```

(Permet de cibler un élément dont la classe finit par 1)

HTML

```
<input class="2padding">
```

CSS

```
[class^="2"] {...}
```

(Permet de cibler un élément dont la classe commence par 2)

HTML

```
<input class="padding">
```

CSS

```
[class*="padding"] {...}
```

(Permet de cibler un élément qui contient la classe padding)

HTML

```
<input class="padding">
```

CSS

```
[class] {...}
```

(Permet de cibler un élément qui contient l'attribut class)

Les sélecteurs

Les combinateurs

Les sélecteurs de voisin direct

CSS

div + div {...}

Permet de cibler n'importe quel élément qui suit immédiatement un élément

Les sélecteurs de voisins

CSS

div ~ div {...}

Permet de cibler n'importe les éléments qui suivent (immédiatement ou non) un élément et qui ont le même parent

Les sélecteurs d'éléments enfants

CSS

ul > li {...}

Permet de cibler tous les éléments <li> qui sont directement situés sous un élément

Les sélecteurs

Les combinateurs

Les sélecteurs d'éléments descendants

CSS

div span {...}

Permet de cibler n'importe quel élément situé à l'intérieur d'un élément

Les sélecteurs multiples

CSS

div, p {...}

Permet de cibler n'importe quels éléments et <p> en même temps

Les sélecteurs

Les pseudo-classes structurelles

nth-child

CSS

div:nth-child(An+B) {...}

A et B sont deux entiers positifs, nuls ou négatifs.
A représente une taille de cycle et B est une valeur de décalage.

Exemple :

div:nth-child(2n+2) {...}

Toutes les 2 balises à partir de la 2ème

Exemple :

div:nth-child(1) {...}

La 1ère balise

Exemple :

div:nth-child(3n+1) {...}

Toutes les 3 balises à partir de la 1ère

Les sélecteurs

Les pseudo-classes structurelles

Exercice 1 :
div:nth-child(1n+1) {...}

Exercice 2 :
div:nth-child(2n+4) {...}

Exercice 3 :
div:nth-child(-3n+7) {...}

Exercice BONUS :
div:nth-child(?) {...}

Les sélecteurs

Les pseudo-classes structurelles

Correction Exercice 1 :
div:nth-child(1n+1) {...}

Correction Exercice 2 :
div:nth-child(2n+4) {...}

Correction Exercice 3 :
div:nth-child(-3n+7) {...}

Correction Exercice BONUS :
div:nth-child(-n+4) {...}

Les sélecteurs

Les pseudo-classes structurelles

nth-last-child

CSS

div:nth-last-child(An+B) {...}

Fonctionne de la même manière que nth-child mais en partant du dernier élément.

Exemple :

div:nth-last-child(2n+1) {...}

Toutes les 2 balises à partir de la dernière balise

Les sélecteurs

Les pseudo-classes structurelles

first-child

CSS

div:first-child {...}

Permet de cibler le premier élément (équivalent à div:nth-child(1))

last-child

CSS

div:last-child {...}

Permet de cibler le dernier élément (équivalent à div:nth-last-child(1))

nth-of-type

CSS

div:nth-of-type(An+B) {...}

Fonctionne de la même manière que nth-child mais ne prend en compte que les éléments de type

Les sélecteurs

Les pseudo-classes structurelles

first-of-type

CSS

div:first-of-type {...}

Permet de cibler le premier élément parmis les éléments de type

last-of-type

CSS

div:last-of-type {...}

Permet de cibler le dernier élément parmis les éléments de type

Les sélecteurs

Les pseudo-classes dynamiques

:link

CSS

a:link {...}

Permet de cibler les éléments <a> qui n'ont pas encore été visités

:visited

CSS

a:visited {...}

Permet de cibler les éléments <a> qui ont été visités

:hover

CSS

button:hover {...}

S'applique au survol de l'élément <button>

Les sélecteurs

Les pseudo-classes dynamiques

:active

CSS

a:active {...}

Permet de cibler les éléments <a> qui ont été activés

:focus

CSS

a:focus {...}

Permet de cibler les éléments <a> qui ont le “focus”

:before

CSS

button:before {...}

Permet d’appliquer un style devant l’élément <button>

Les sélecteurs

Les pseudo-classes dynamiques

:after

CSS

button:after {...}

Permet d'appliquer un style après l'élément <button>

03

Les fondamentaux du web (HTML, CSS, PHP)

Partie 2 - Le CSS

Les propriétés

Les propriétés

Le positionnement

position

static

position: static

Par défaut, un bloc ( ) est en position statique. C'est-à-dire qu'il est positionné dans le flux de la page => par rapport aux contenus présents. Ainsi, il n'est pas possible de le déplacer par défaut (théoriquement).

relative

position: relative

L'élément est positionné par défaut dans le flux. Mais contrairement au positionnement statique, nous pouvons modifier un élément qui est en positionnement relatif grâce aux propriétés top, right, bottom et left.
À savoir que le déplacement d'un élément possédant cette propriété n'impact pas la position des autres éléments de la page.

Les propriétés

Le positionnement

absolute

position: absolute

L'élément est retiré du flux normal et aucun espace n'est créé pour l'élément sur la page. Il est ensuite positionné par rapport à son parent le plus proche positionné s'il y en a un ou par rapport au bloc englobant initial sinon. La position finale de l'élément est déterminée par les valeurs de top, right, bottom et left.

fixed

position: fixed

Comme pour la position absolue, l'élément est retiré du flux normal et aucun espace n'est créé pour l'élément sur la page.
Par rapport aux autres positionnements, un bloc fixe est positionné par rapport à la fenêtre.
Ainsi, il est toujours au même endroit, même si la page défile.
La position finale de l'élément est déterminée par les valeurs de top, right, bottom et left.

Les propriétés

Le positionnement

sticky

position: sticky

L'élément est positionné en fonction de la position de défilement de l'utilisateur.
Un élément collant bascule entre relative et fixed, en fonction de la position de défilement.

Les propriétés

Largeur et hauteur

Les unités

Avant de parler de la taille des éléments, il est indispensable d'inclure la notion d'unités.

En effet, en CSS, il existe plusieurs unités pour exprimer les dimensions.
Il y a des unités absolues : pixels (px), pouces (in), centimètres (cm)...
Il y a également des unités relatives au texte. La plus connue est rem qui fait référence à la taille de la police de l'élément racine.
Nous utiliserons principalement px et rem.

Pour certaines propriétés, il est possible d'utiliser % pour calculer en fonction du bloc parent.

Les propriétés

Largeur et hauteur

width

width: ...px

Cette propriété permet de définir la largeur de la boîte du contenu d'un élément. Par défaut, sa valeur est auto, c'est-à -dire la largeur automatiquement calculée par rapport à son contenu.

height

height: ...px

Cette propriété permet de définir la hauteur de la boîte du contenu d'un élément. Par défaut, sa valeur est auto, c'est-à -dire la hauteur automatiquement calculée par rapport à son contenu.

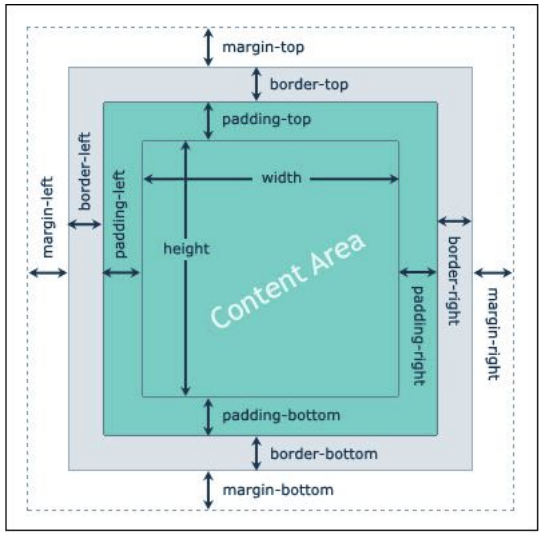
Les propriétés

Largeur et hauteur

Comme dit auparavant, un élément possède une largeur et hauteur par défaut, qui se fait par rapport au contenu de cet élément.

Cependant, la “véritable” largeur d’un bloc est l’addition de plusieurs propriétés.

Pour visualiser la largeur d’un élément, il faut donc prendre en compte ce schéma :



Les propriétés

Les marges internes

padding

padding : ...px

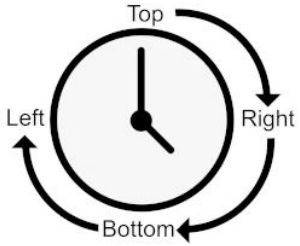
Cette propriété correspond à l'espace entre le contenu de l'élément et sa bordure. Les valeurs négatives ne sont pas autorisées.

La propriété padding synthétise les propriétés padding-top, padding-right, padding-bottom, padding-left.

```
div {
  padding: 10px;
}
```

→

```
div {
  padding-top: 10px;
  padding-right: 10px;
  padding-bottom: 10px;
  padding-left: 10px;
}
```



Les propriétés

Les bordures

border

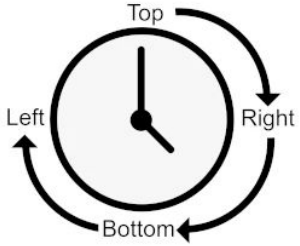
border : largeur | style | couleur *OU* largeur | style *OU* style | couleur *OU* style

La propriété CSS border est une propriété raccourcie qui permet de définir les propriétés liées à la bordure. border peut être utilisée pour définir les valeurs de border-width, border-style et border-color.

```
div {
  border: 2px solid black;
}
ou
div {
  border-width: 2px;
  border-style: solid;
  border-color: black;
}
```

→

```
div {
  border-top: 2px solid black;
  border-right: 2px solid black;
  border-bottom: 2px solid black;
  border-left: 2px solid black;
}
```



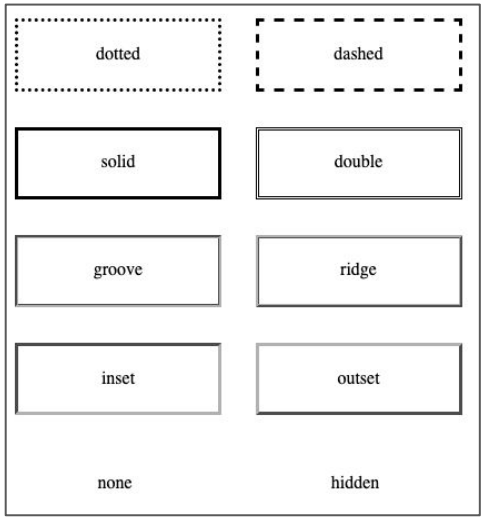
Les propriétés

Les bordures

border-style

border-style : ...

La propriété CSS border-style permet de définir le style des lignes utilisées pour les bordures des quatre côtés de la boîte d'un élément.



Les propriétés

Les bordures

border-radius

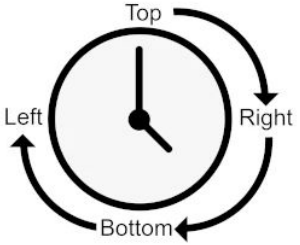
border-radius : ...px

La propriété CSS border-radius permet de définir des coins arrondis pour la bordure d'un élément.
La propriété border-radius synthétise les propriétés border-top-left-radius, border-top-right-radius, border-bottom-left-radius, border-bottom-right-radius.

```
div {
  border-radius: 50px;
}
```

→

```
div {
  border-top-left-radius: 50px;
  border-top-right-radius: 50px;
  border-bottom-right-radius:50px;
  border-bottom-left-radius: 50px;
}
```



Les propriétés

Les marges externes

margin

margin: ...px

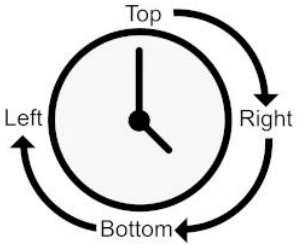
Cette propriété correspond à la taille des marges sur les quatre côtés de l'élément. Contrairement à la propriété padding, il est possible d'utiliser des valeurs négatives.

La propriété margin synthétise les propriétés margin-top, margin-right, margin-bottom, margin-left.

```
div {
  margin: -25px;
}
```

→

```
div {
  margin-top: -25px;
  margin-right: -25px;
  margin-bottom: -25px;
  margin-left: -25px;
}
```



Les propriétés

Les types d'affichage

display

Définit le type d'affichage utilisée pour le rendu d'un élément

Commençons par les types d'affichage dit "outer" c'est-à-dire le type d'affichage de l'élément en soi et son comportement visuel par rapport aux autres.

none

display: none

L'élément n'est pas affiché et sort du flux.

inline

display: inline

L'élément est interprété comme étant un seule ligne. Les éléments inline vont venir essayer de se placer en ligne, à côté de l'élément qui les précèdent.
Un élément de type inline ne peut contenir un élément de type block.

Les propriétés

Les types d'affichage

block

display: block

L'élément est interprété comme un bloc, et prend ainsi toute la largeur disponible au sein de son élément parent.
Un élément en block prendra automatiquement un ligne, il ne se positionne jamais à côté d'un élément.

inline

display: inline

L'élément est interprété comme étant un seule ligne. Les éléments inline vont venir essayer de se placer en ligne, à côté de l'élément qui les précèdent.
Un élément de type inline ne peut contenir un élément de type block.

Passons maintenant aux types d'affichage dit "inner" c'est-à-dire le type d'affichage du contenu à l'intérieur d'un élément.

Les propriétés

Les types d'affichage

flex

display: flex

L'élément est interprété comme un bloc par rapport aux autres éléments de la page, mais l'intérieur va se construire de façon à distribuer l'espace entre les blocs.

flex-direction

flex-direction: ...

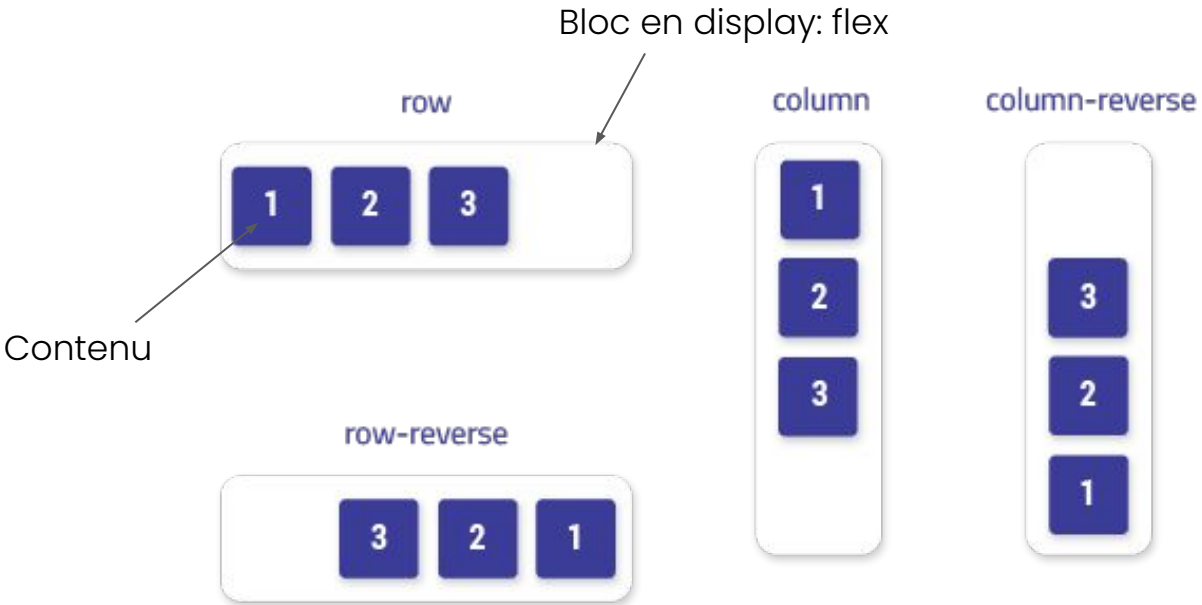
Par défaut, la valeur flex-direction, lorsque la propriété n'est pas définie, est row.
Il existe 4 valeurs pour cette propriété : row | row-reverse | column | column-reverse.

La valeur flex de la propriété display et la propriété flex-direction sont liées car cela définit la façon dont les éléments flexibles sont placés dans un conteneur flexible.

Les propriétés

Les types d'affichage

Exemple :



Les propriétés

Les types d'affichage

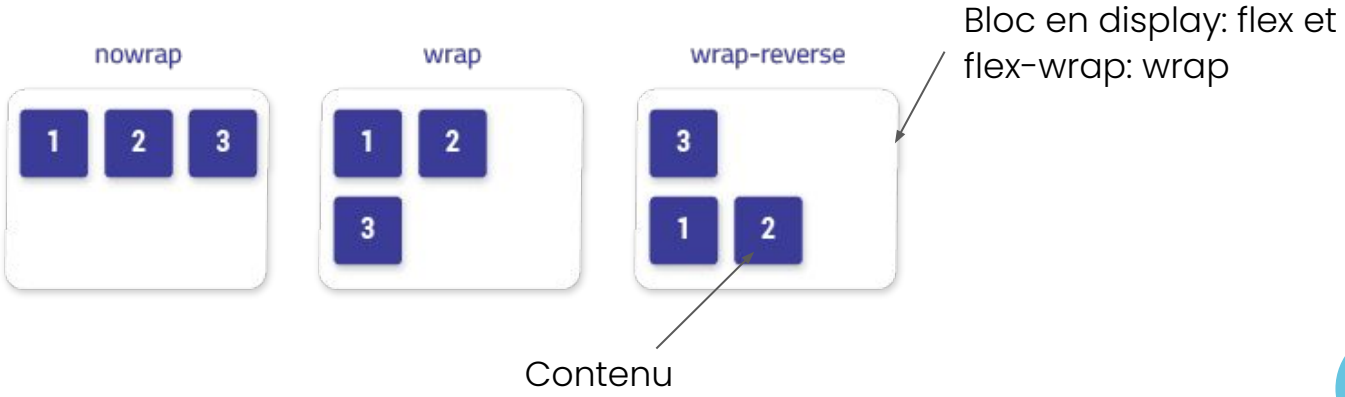
La propriété flex-direction n'est pas la seule qui est liée au display: flex, il y aussi :

flex-wrap

flex-wrap: ...

Indique si les éléments flexibles sont contraints à être disposés sur une seule ligne ou s'ils peuvent être affichés sur plusieurs lignes avec un retour automatique.

Il existe 3 valeurs pour cette propriété : nowrap | wrap | wrap-reverse.



Les propriétés

Alignement des conteneurs flexibles

Maintenant que nous avons vu la création de conteneurs flexibles, nous allons voir comment les aligner sur notre page.

align-items

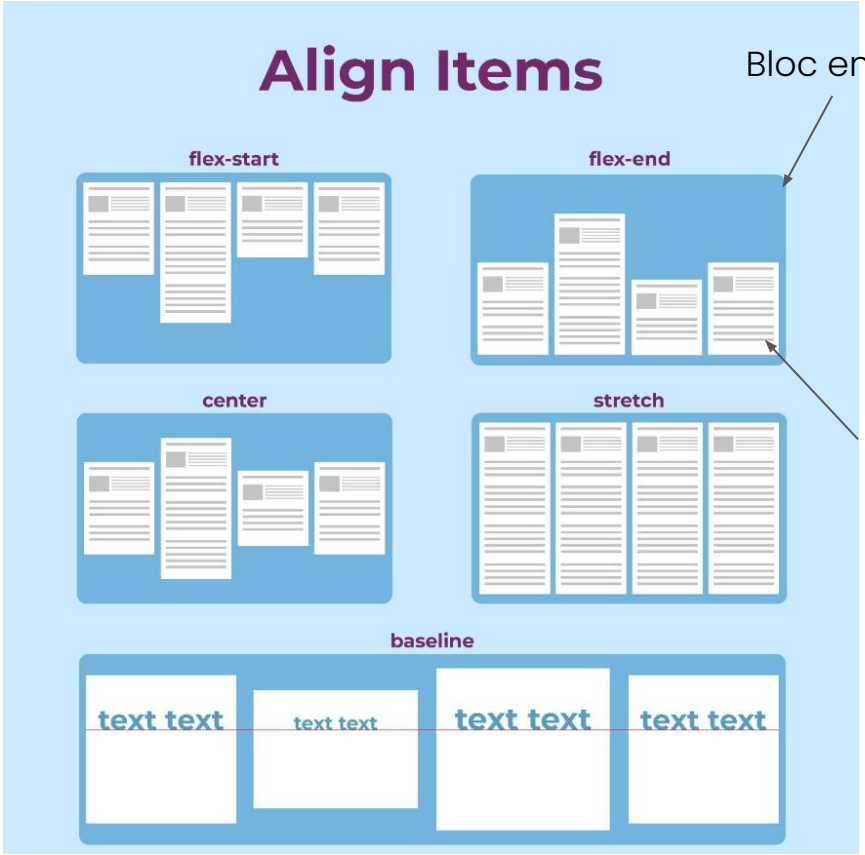
align-items: ...

Spécifie l'alignement par défaut des éléments à l'intérieur d'une boîte flexible ou d'un conteneur de grille.

Il existe plusieurs valeurs pour cette propriété : normal | stretch | center | flex-start | flex-end | start | end | baseline | initial.

Les propriétés

Alignement des conteneurs flexibles



Les propriétés

Alignement des conteneurs flexibles

Maintenant que nous avons vu la création de conteneurs flexibles, nous allons voir comment les aligner sur notre page.

justify-content

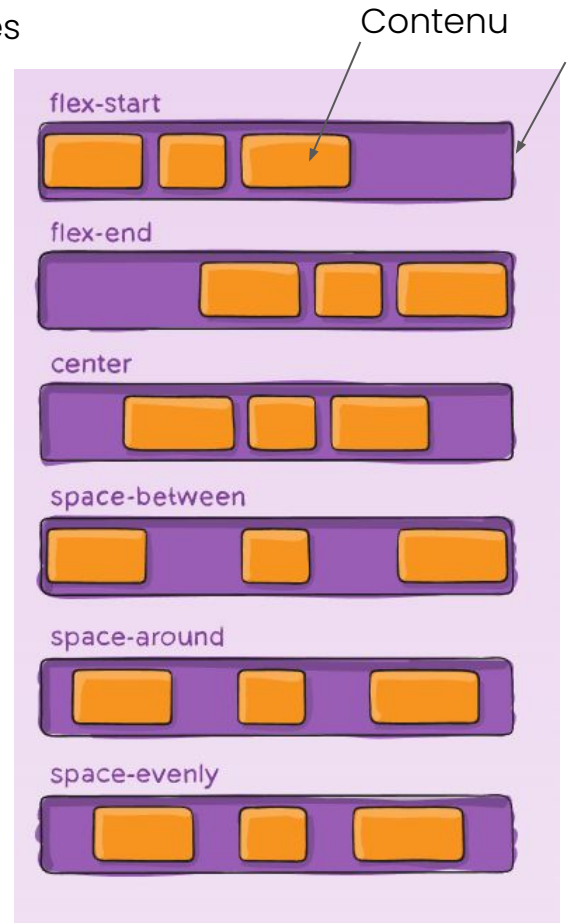
justify-content: ...

Aligne les éléments du conteneur flexible lorsque les éléments n'utilisent pas tout l'espace disponible sur l'axe principal (horizontalement).

Il existe plusieurs valeurs pour cette propriété : flex-start | flex-end | center | space-between | space-around | space-evenly | initial.

Les propriétés

Alignement des conteneurs flexibles



Les propriétés

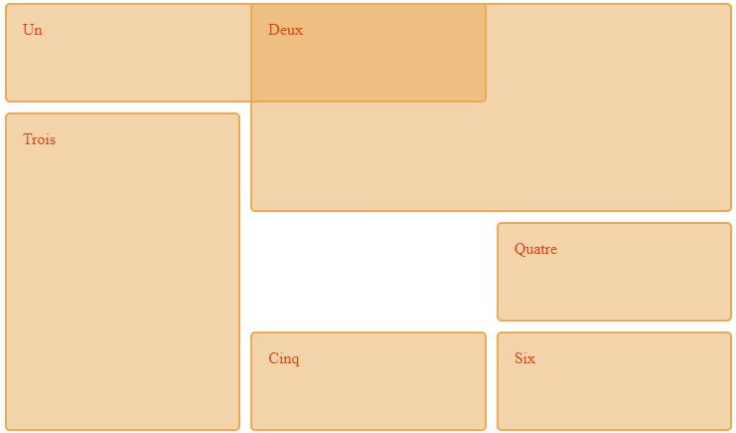
Les types d'affichage

grid

display: block

Cette propriété va diviser notre bloc / page en tableau afin de positionner les éléments sous forme de colonnes et de lignes.
Cependant, contrairement aux tableaux, nous n'aurons pas de structures de contenu.

Exemple :



Les propriétés

Les conteneurs en grille

Il existe plusieurs propriétés afin de gérer les conteneurs en grille.

grid-template-columns

grid-template-columns: ...px | ...fr | ...ch ..... (par colonne)

Cette propriété va définir la taille des colonnes de la grille.

grid-template-rows

grid-template-rows: ...px | ...fr | ...ch ..... (par ligne)

Cette propriété va définir la taille des lignes de la grille.

Particularité : Si vous voulez créer plusieurs colonnes / lignes de même taille, il faudra utiliser la propriété comme ceci :
grid-template-columns: repeat(3, 1fr);

Les propriétés

Les conteneurs en grille

grid-column-start

grid-column-start: n (1, 2, 3, 4....)

Définit la position du début d'un élément de la grille.

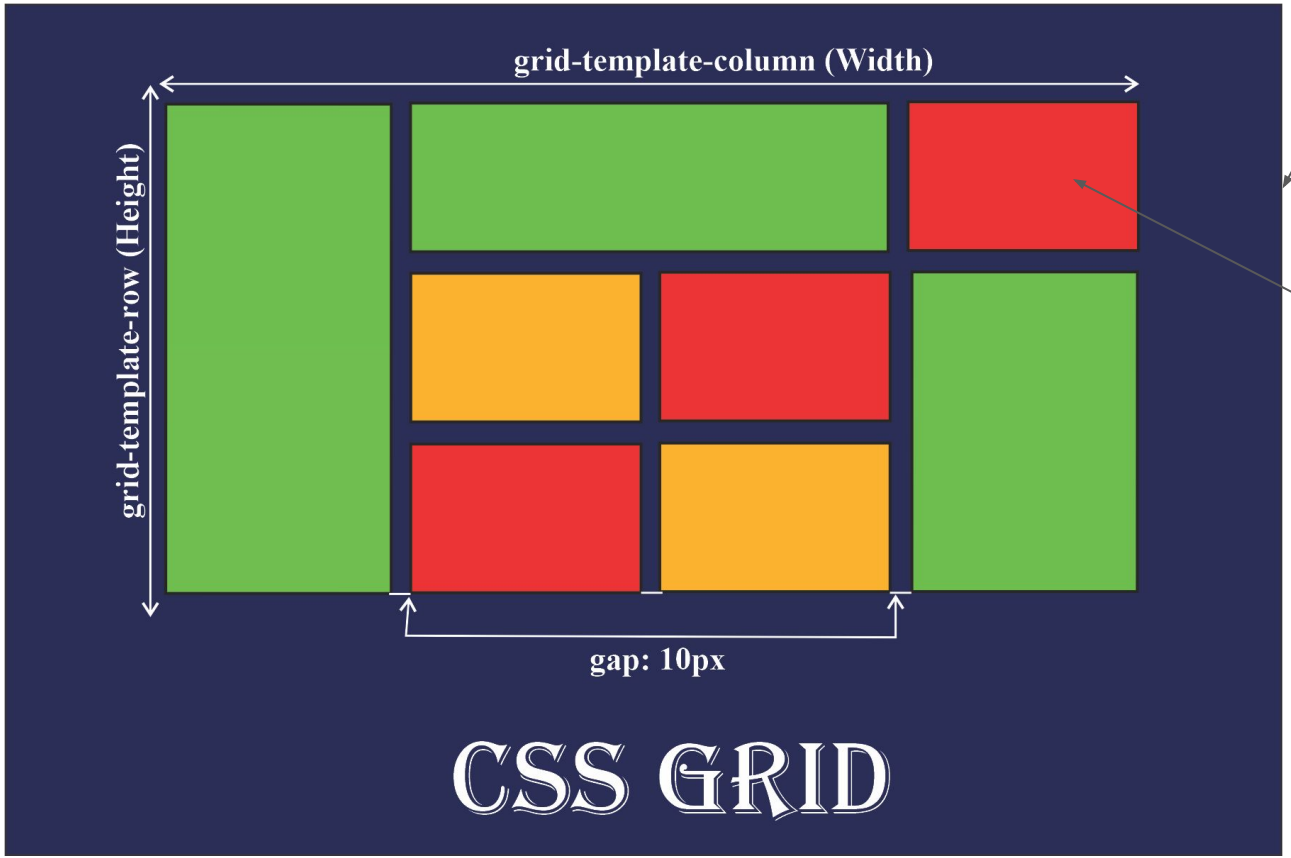
grid-column-end

grid-column-end: n (1, 2, 3, 4....)

Définit la position de la fin d'un élément de la grille.

Les propriétés

Les conteneurs en grille



Les propriétés

Les gouttières

column-gap

column-gap : ...px | ...rem

Spécifie l'écart entre les colonnes dans une grille, une boîte flexible ou une disposition à plusieurs colonnes.

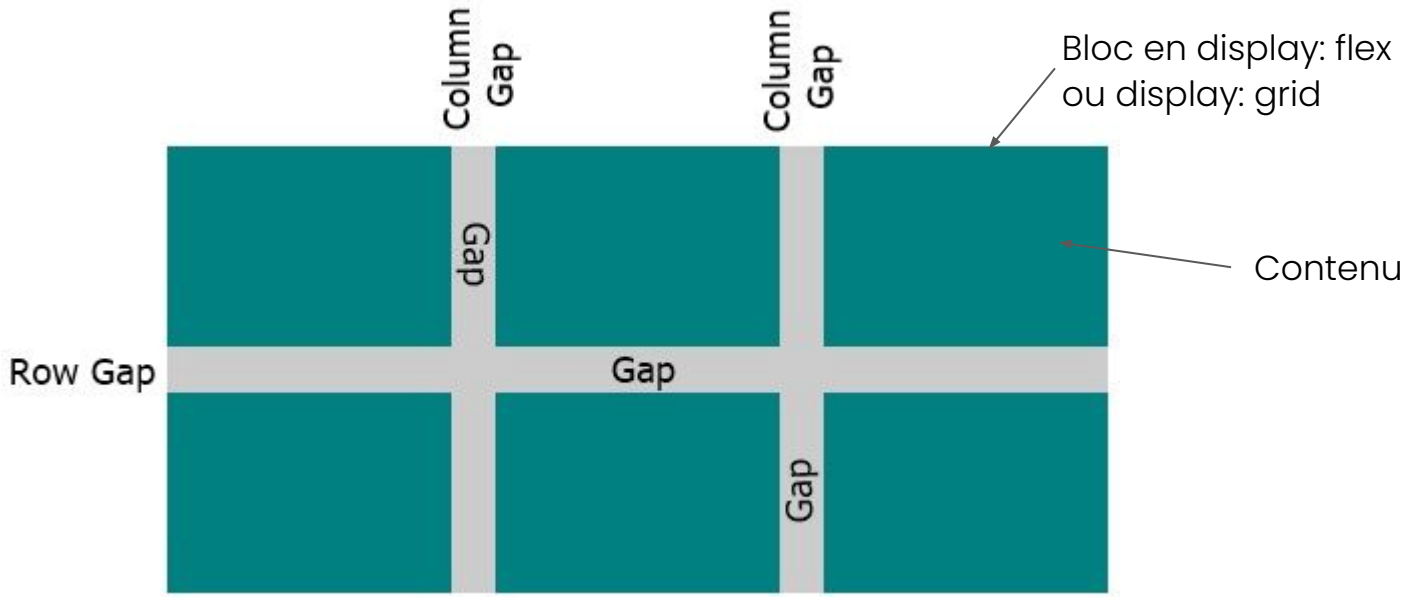
row-gap

row-gap : ...px | ...rem

Spécifie l'écart entre les lignes dans une grille, une boîte flexible ou une disposition à plusieurs lignes.

Les propriétés

Alignement des conteneurs flexibles



Les propriétés

Les couleurs

Les couleurs en CSS peuvent être définies de plusieurs manières.

Les couleurs prédéfinies

Par défaut, il est possible d'utiliser une des 132 couleurs définis de base dans le CSS.

Exemple : *color : red;*

Le code RGB avec des valeurs décimales

Il est possible de définir une couleur grâce à son code RGB. Il existe 256 niveaux différents pour chacune de ces trois couleurs, soit 16 777 216 couleurs différentes (256^3).

En décimal, on utilise le mot-clé rgb suivi entre parenthèses du niveau choisi (de 0 à 255) pour les trois couleurs.

Exemple : *color: rgb(255, 0, 0);* (ce qui signifie 255 pour le rouge, 0 pour le vert et 0 pour le bleu)

Les propriétés

Les couleurs

Le code RGB avec des valeurs hexadécimales

En hexadécimal, on utilise le mot-clé # suivi du niveau choisi pour les trois couleurs. Chaque couleur peut être représentée par un caractère hexadécimal (de 0 à F) ou par deux caractères hexadécimaux (de 00 à FF). Le code RGB peut donc être composé de trois caractères permettant de coder 4096 couleurs ou de six caractères permettant de coder 16 777 216 couleurs. Le code couleur#F08 équivaut au code couleur #FF0088.

Exemple : *color: #FF0000;* (ce qui signifie FF pour le rouge, 00 pour le vert et 00 pour le bleu)

Les propriétés

Les arrières-plans

background-color

background-color : color

Cette propriété permet de définir la couleur utilisée pour l'arrière-plan d'un élément.

background-image

background-image: url(“….”)

Cette propriété permet de définir une ou plusieurs images comme arrière(s)-plan(s) pour un élément.

Exemple : *background-image: url("/Images/image.png");*

Les propriétés

Les propriétés “textuelles”

Le CSS possède également la capacité de formater des textes.

color

color : color

Cette propriété permet de définir la couleur d’un élément de type texte.

text-align

text-align: ...

Cette propriété permet de définir l’alignement d’un texte dans un bloc / conteneur.

Il existe plusieurs valeurs pour cette propriété : inherit | center | left | right | justify.

Les propriétés

Les propriétés “textuelles”

text-decoration

text-decoration : ...

Cette propriété permet de définir des “décorations” pour un texte, comme le soulignement par exemple.

Il existe plusieurs valeurs pour cette propriété : inherit | none | underline | overline | line-through.

text-transform

text-transform: ...

Cette propriété permet de contrôler la capitalisation du texte.

Il existe plusieurs valeurs pour cette propriété : none | capitalise | uppercase | lowercase | inherit.

Les propriétés

Les propriétés “textuelles”

text-indent

text-indent : ...px | ...rem

Cette propriété permet de définir une indentation / alinéa au début du texte en question.

letter-spacing

letter-spacing : ...px | ...rem

Cette propriété est utilisée pour spécifier l'espace entre les caractères dans un texte.

line-height

line-height : ...px | ...rem

Cette propriété est utilisée pour spécifier l'espace entre les lignes.

Les propriétés

Les propriétés “textuelles”

text-shadow

text-shadow: ...px | ...rem (Ombre horizontale) ...px | ...rem (Ombre verticale)

Cette propriété est utilisée pour ajouter une ombre à un texte.

font-style

font-style : ...

Cette propriété est principalement utilisée pour spécifier du texte en italique.

Il existe plusieurs valeurs pour cette propriété : normal | italic | oblique.

font-size

font-size : ...px | ...rem

Cette propriété définit la taille du texte.

Les propriétés

Les propriétés “textuelles”

font-family

font-family: ...

Cette propriété est utilisée pour spécifier la police d'un texte.
Remarque : Si le nom de la police est composé de plusieurs mots, il doit être entre guillemets, comme : "Times New Roman".

Exemple : *font-family: "Times New Roman", serif;*

Les propriétés

Autres propriétés utiles

transition

transition : ...

Les transitions CSS permettent de modifier les valeurs des propriétés en douceur, sur une durée donnée.

Exemple : *transition: background 0.2s ease;*
Ici, nous aurons une transition sur le background de notre élément avec l'effet "ease" pendant 0.2 secondes.

opacity

opacity: 0 à 1

Cette propriété définit le niveau d'opacité d'un élément.

03

Les fondamentaux du web (HTML, CSS, PHP)

Partie 2 - Le CSS

Mobile-First & Responsive

Mobile-first & Responsive

Il est habituel pour les développeurs de créer des sites qui soient autant accessibles sur Desktop que sur Mobile.

Ainsi, depuis quelques années, les bonnes pratiques du web veulent que les développeurs créent leurs sites en “Mobile-first”, c'est-à-dire que l’on développe du support mobile vers le support Desktop.

Mais comment faire cela ?

Mobile-first & Responsive

Les médias-queries

Les médias-queries ne sont pas des propriétés CSS comme nous venons de le voir, mais des “requêtes” (le mot importe peu) qui permettent de déclarer des propriétés pour un élément donné pour **UNE RÉSOLUTION PARTICULIÈRE.**

Exemple :

```
@media all and (max-width: 700px) {
        div {
                width: 100%;
                margin-right: 0;
        }
        #menu > div {
                width: 100px;
                margin-right: 10px;
        }
        .content {
                width: 100%;
                margin-top: 0;
        }
}
```

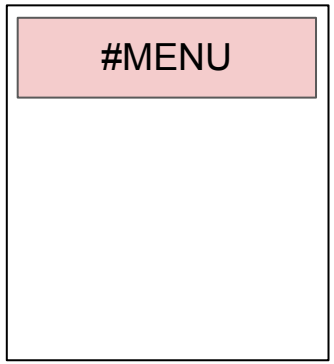
Ici ce qui nous intéresse est la valeur entre parenthèses.

Si nous voulons que des propriétés s’activent après une certaine résolution, nous mettrons *min-width* et inversement, si nous voulons que cela s’active avant une certaine résolution, nous mettrons *max-width*.

Mobile-first & Responsive

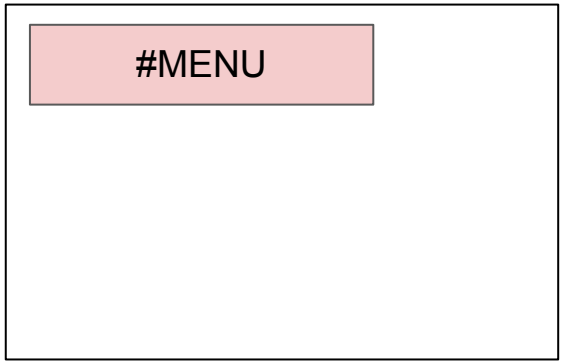
Les médias-queries

Exemple :



Nous sommes sur une tablette de 728px de large.

Nous voulons que notre menu ne fasse plus toute la largeur de notre écran après 1024px de large.



Réponse :

```
@media all and (min-width: 1024px) {
        #menu  {
                width: 70%;
        }
}
```