

R111 Intégration

2) Bases de CSS



Préambule

- En intégration, on sépare généralement le fond et la forme :
 - **HTML** permet de décrire la **structure** et le **contenu** d'une page web
 - **CSS** décrit la **mise en forme** de la page
- Cette séparation permet de changer le style d'un site sans modifier son contenu
ex : <http://www.csszengarden.com>
- Tutoriel CSS : <https://developer.mozilla.org/fr/docs/Learn/CSS>

Contenu

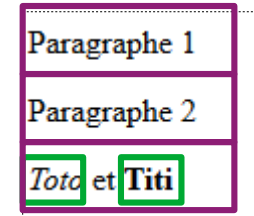
- Éléments *block* et *inline* en HTML
- CSS : Cascading Style Sheet (feuilles de style en cascade)
 - Règles CSS :
 - Sélecteurs
 - Propriétés
 - Fontes
 - Couleurs
 - Bordures
 - Dimensions
- Cascade, héritage, et spécificité

Deux types d'éléments HTML

- Éléments de type ***block*** :
 - S'affichent les uns en dessous des autres
 - Ex : titres, paragraphes, éléments de liste, tableaux...
- Éléments de type ***inline*** :
 - S'affichent les uns à la suite des autres (côte à côte)
 - Ex : liens, emphase, renforcement...

Exemple

```
<p>Paragraphe 1</p>  
<p>Paragraphe 2</p>  
<p><em>Toto</em> et <strong>Titi</strong></p>
```



2 éléments *inline*

3 éléments *block*

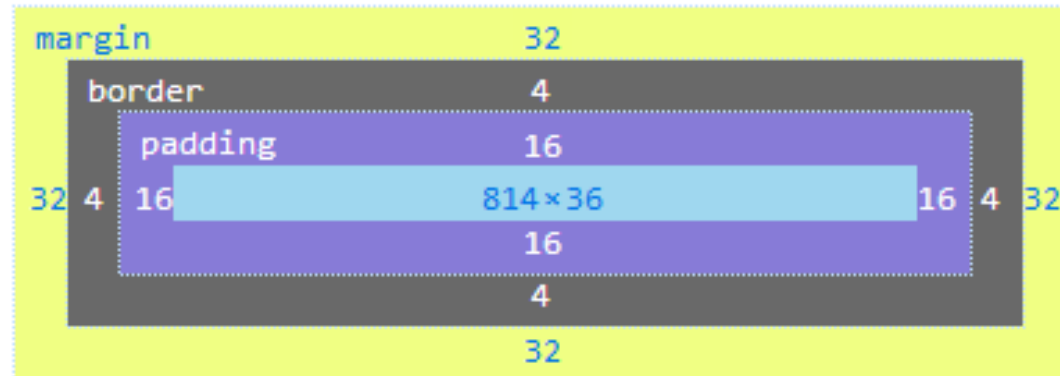


Les éléments *inline*

- Ont la taille de leur contenu :
 - Impossible de les redimensionner
- Ne peuvent contenir que des éléments de type *inline*

Les éléments *block*

- Ont des dimensions : largeur (***width***) et hauteur (***height***)
- Peuvent contenir d'autres éléments (de type bloc et/ou inline)
- Possèdent des marges internes (***padding***) et externes (***margin***)



Les éléments HTML neutres

- Vous avez vu jusqu'ici des éléments HTML qui ont un sens (ceci est un titre, ceci est un texte important...)
- On a parfois besoin d'élément **sans aucun sens particulier**.

Dans ce cas, on utilise :

- `<div>` pour un **élément neutre de type *block***
- `` pour un **élément neutre de type *inline***



CSS

Cascading Style Sheet

CSS ?

- C'est le langage de style, de mise en forme
- Toujours associé à du HTML
- En CSS, on définit des **règles** de style qui vont s'appliquer aux éléments HTML
- Toutes les propriétés d'un élément HTML sont modifiables
- Trois façons d'utiliser CSS :
 - Dans l'entête du document (balise <style> dans l'élément <head>) 🙄
 - Dans l'attribut style de l'élément à modifier 😡
 - En liant le document HTML à un fichier CSS 😊

Stylez !

```
<head>
  <title>Ma page</title>
  <style> Des règles CSS ici :-( </style>
</head>
<body>
  <h1 style="color : blue">Grand titre</h1>
</body>
```



```
<head>
  <title>Ma page</title>
  <link rel="stylesheet" href="chemin/vers/fichier.css">
</head>
<body>
  <h1>Grand titre</h1>
</body>
```





Les règles CSS

Les règles CSS

- Une règle CSS comprend :
 - Un **sélecteur** permettant de choisir quels éléments vont être concernés par la règle
 - Un groupe d'accolade { ... } qui contient une ou plusieurs "propriété : valeur;"
 - La **propriété** définit ce que l'on veut modifier (police, couleur, taille, etc.)
 - La **valeur** est la nouvelle valeur qui sera appliquée à la propriété
- Exemple :

Sélecteur h1 : la règle va modifier le style des titres de niveau 1

```
h1 {  
  color : blue;  
  font-style : italic;  
}
```

Propriétés color et font-style : la règle va modifier les couleurs et la police



Sélecteurs CSS

https://developer.mozilla.org/fr/docs/Learn/CSS/Building_blocks/Selectors

Sélecteurs CSS

- Ils permettent de sélectionner le(s) élément(s) concerné(s) par une règle CSS

- Les sélecteurs simples :

- Sélection par le **type d'élément** :

h3 : tous les <h3>

- Sélection par l'**id** :

#mon-id : l'élément avec id="mon-id"

- Sélection par la **classe** :

.ma-classe : les éléments ayant "ma-classe"

A noter :

Il est possible de combiner ces sélecteurs. Par exemple :

- **p.red** : les p ayant la classe 'red'
- **h2#titre.news** : le h2 d'id 'titre' qui a la classe 'news'

Sélecteurs CSS

- Regroupement de sélecteurs avec ','

```
h1, h2, .rouge, #titre {  
    color : red ;  
}
```

Les titres de niveau 1 et 2, les éléments ayant la classe 'rouge' et l'élément d'id 'titre' seront en rouge

- Sélecteur universel avec '*'

```
* {  
    margin : 0 ;  
}
```

Aucune marge pour tous les éléments

Sélecteurs CSS

- Problème :
 - Avec les sélecteurs vu jusqu'ici, comment sélectionner l'un des titres h3 mais pas l'autre ?

```
<div id="gauche">  
  <h3>titre de gauche</h3>  
</div>  
<div id="droite">  
  <h3>titre de droite</h3>  
</div>
```

- Solution : utiliser les combinateurs
 - Ils combinent plusieurs sélecteurs. Par exemple : sélectionner un élément 'h3' qui se trouve dans un élément '#droite'

Combinateurs

- **Combinateur de descendance**

- S'écrit avec un **espace** ' '
- Sélectionne tous les **éléments qui descendent d'un autre**
- Par exemple : sélectionner tous les 'h3' descendant d'un div#droite

```
div#droite h3
```

- **Combinateur enfant**

- Se note avec un '>'
- Sélectionne les **enfants directs d'un élément**

```
div#droite > h3
```

Différence entre les combinateurs ' ' et '>'

```
<ul id="menu">
  <li>Item 1</li>
  <li>Item 2</li>
  <li>Sous-menu
    <ul>
      <li>Item 3.1</li>
      <li>Item 3.2</li>
      <li>Item 3.3</li>
    </ul>
  </li>
</ul>
```

`ul#menu li`

Sélectionne 6 éléments

`ul#menu > li`

Sélectionne 3 éléments

Combinateurs de frères (*siblings*)

- **Combinateur de frère**
 - Se note avec un '~'
 - Sélectionne les **éléments de même niveau hiérarchique qu'un autre élément**
- **Combinateur de frère adjacent**
 - S'écrit avec un plus '+'
 - Sélectionne tous les **éléments de même niveau hiérarchique qui suivent un autre élément**

Combinateurs '+' et '~'

```
<ul>  
  <li>Fraise</li>  
  <li>Banane</li>  
  <li>Abricot</li>  
  <li id="peach">Pêche</li>  
  <li>Pastèque</li>  
  <li>Melon</li>  
</ul>
```

`li#peach ~ li` Sélectionne 5 éléments

`li#peach + li` Sélectionne 2 éléments

Sélecteurs d'attributs

- On peut aussi sélectionner des éléments par leurs attributs

[attr]	Sélectionne les éléments ayant un attribut 'attr'
[attr=valeur]	Sélectionne les éléments dont l'attribut 'attr' vaut exactement 'valeur'
[attr~=valeur]	Sélectionne les éléments dont l'attribut 'attr' contient exactement le mot 'valeur' séparé d'autres valeurs par des espaces
[attr*=valeur]	Sélectionne les éléments dont l'attribut 'attr' contient au moins une occurrence du mot 'valeur'
[attr^=valeur]	Sélectionne les éléments qui possèdent un attribut 'attr' dont la valeur commence par valeur.
[attr\$=valeur]	Sélectionne les éléments qui possèdent un attribut 'attr' dont la valeur se termine par valeur.

Pseudo-classes

- Les pseudo-classes permettent de sélectionner des éléments lorsqu'ils se trouvent dans un état particulier

<code>:hover</code>	L'élément lorsqu'il est survolé par le curseur
<code>:focus</code>	L'élément lorsqu'il a le <i>focus</i>
<code>:link</code>	Les liens non visités
<code>:visited</code>	Les liens visités
<code>:first-child</code>	L'élément qui est le premier enfant de son parent
<code>:first-of-type</code>	L'élément qui est le premier enfant de son type parmi tous ses frères

Il en existe beaucoup d'autres, voir la doc.

Par exemple `:last-child` `:last-of-type` `:nth-child` `:nth-of-type` `:valid` ...

Pseudo-éléments

- Les pseudo-éléments permettent de sélectionner une partie d'un élément

::first-letter	Première lettre d'un élément
::first-line	Première ligne d'un élément
::selection	La partie de l'élément sélectionnée à la souris
::before	Crée un pseudo élément qui sera le premier enfant
::after	Crée un pseudo élément qui sera le dernier enfant

Quelques exemples

`h2 + p::first-letter`

La première lettre du paragraphe qui suit un `<h2>`

`article a:visited`

Tous les liens visités dans un `<article>`

`ul > li > ul`

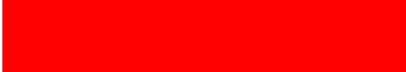



Les sous-listes



Propriétés CSS

Couleurs

- On utilise les propriétés :
 - **color** pour la couleur d'avant plan (le texte)
 - **background-color** pour l'arrière plan
- Plusieurs format de valeur possibles :
rgb(), rgba(), hexadécimal, mots clés

Couleur	Hexadécimal	RGB	Mot clé
	#FF0000	rgb(255,0,0)	red
	#00FF00	rgb(0,255,0)	reen
	#0000FF	rgb(0,0,255)	blue
	#D2691E	Rgb(210,105,30)	chocolate
	#7FFF00	Rgb(127,255,0)	chartreuse

Les propriétés pour le texte

- Propriétés pour la police d'écriture :
 - font-family
 - font-size
 - font-style
 - font-weight
- Propriétés pour le texte
 - text-align
 - text-decoration
 - line-height

Propriété font-family

- Définit la famille de police de caractères
- Plusieurs valeurs possibles :
 - Nom de police : `arial`, `times new roman`...
 - Type de police :
 - ♦ `serif` : police avec empattement
 - ♦ `sans-serif` : sans empattement
 - ♦ `monospace` : police à chasse fixe
 - ♦ `cursive` : police cursive
- En pratique, on choisit une liste de plusieurs polices



Mettre entre guillemets si le nom de police contient des espaces



La police souhaitée n'est pas forcément disponible sur la machine de vos utilisateurs

```
font-family: arial, 'Trebuchet MS', helvetica, sans-serif;
```

Propriété font-size

- Définit la taille des caractères
- Plusieurs valeurs possibles :
 - Taille absolue : 12px
 - Taille relative : 1.5em ou 1.25rem
 - Taille prédéfinie : xx-small, x-small, small, medium, x-large...
 - Un pourcentage : 200% pour doubler la taille, 75% pour rapetisser...
- Exemple :



En général, il vaut mieux éviter les tailles absolues

```
font-size: 0.75em;
```

Réduit la taille de police. La nouvelle taille est 3/4 de l'ancienne.

Propriétés font-style et font-weight

- font-style définit le style de la police
- Valeurs possibles :
 - normal, italic, oblique
- font-weight définit la "graisse" de la police
- Valeurs possibles :
 - normal, bold
 - bolder, lighter (pour augmenter ou réduire la graisse)
 - Valeur numérique entre 100 et 900

Les propriétés concernant le texte

- `text-align` : modifie l'alignement du texte
 - Valeurs possibles : `left`, `right`, `center`, `justify`
- `text-decoration` : ajout de soulignement, surlignement...
 - Valeurs possibles : `underline`, `overline`, `none`...
- `line-height` : définit la hauteur d'une ligne de texte
 - Valeurs possibles : une longueur ou un pourcentage

L'arrière plan

- **background-color** définit la couleur d'arrière plan
- On peut mettre une image en arrière plan avec :

```
background-image : url("../chemin/vers/image.jpeg") ;
```

- Quelques possibilités supplémentaires avec les propriétés :
 - **background-repeat** : répéter l'image
 - **background-attachement** : l'image doit elle rester fixe ou "scroller"
 - **background-position**

Dimensionnement

- Les éléments de type *block* peuvent être dimensionnés avec :
 - width, min-width, max-width
 - height, min-height, max-height
- Valeurs possibles :
 - Une longueur (un nombre + une unité)
 - Un pourcentage
 - auto (pour la largeur uniquement)



A noter : il est possible de modifier le type d'élément (*inline* ou *block*) à l'aide de la propriété `display`

La propriété `display`

- La propriété `display` permet de modifier le type d'affichage des éléments
- Valeurs possibles :
 - **block** : pour que l'élément devienne de type block
 - **inline** : l'élément devient de type inline
 - **none** : l'élément n'est plus affiché
 - **inline-block** : l'élément reste placé en ligne, mais il est redimensionnable

Les longueurs en CSS

- Une longueur est un nombre suivi d'une unité (sauf la longueur 0 qui n'a pas besoin d'unité)
- L'unité peut être :
 - Une unité absolue : pixel **px**, centimètre **cm**, point **pt** ...
 - Une unité relative :
 - **em** : hauteur d'un caractère 'M' dans la police actuelle
 - **rem** : taille de police racine
 - **vh** : 1/100 de la hauteur du viewport
 - **vw** : 1/100 de la largeur du viewport



C'est rare d'avoir besoin des unités absolues. Privilégiez les unités relatives

Les bordures

- Une bordure est définie par :

- Son épaisseur : border-width
- Son style : border-style
- Sa couleur : border-color



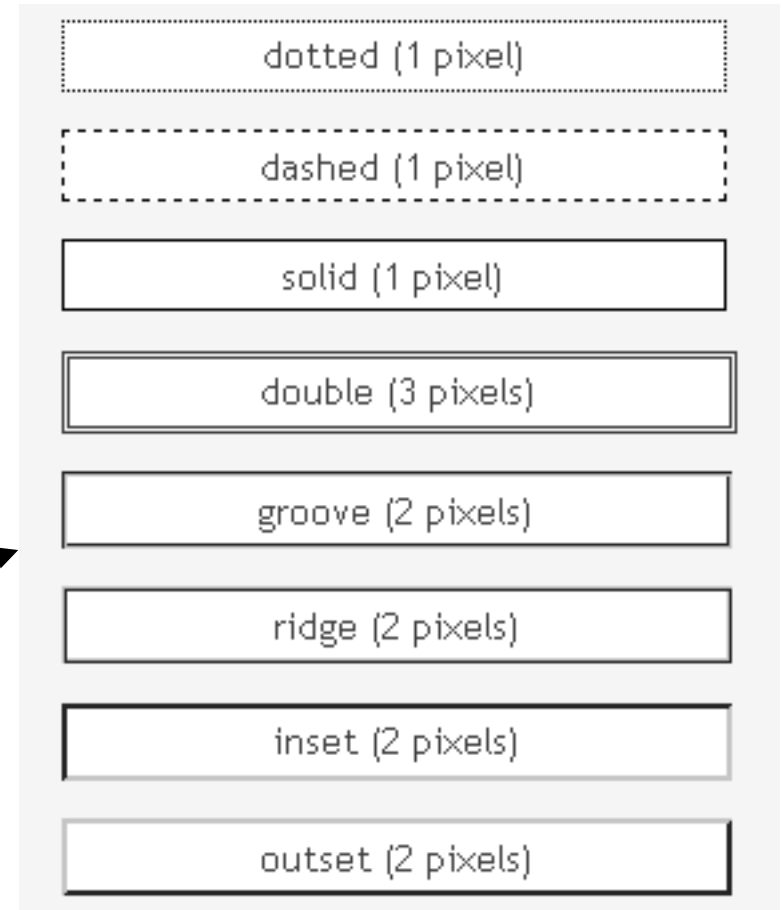
Pensez bien à définir les trois.

- On peut utiliser la propriété raccourcie border
Par exemple :

```
border : 1.5em dotted #FF00FF ;
```

Les bordures

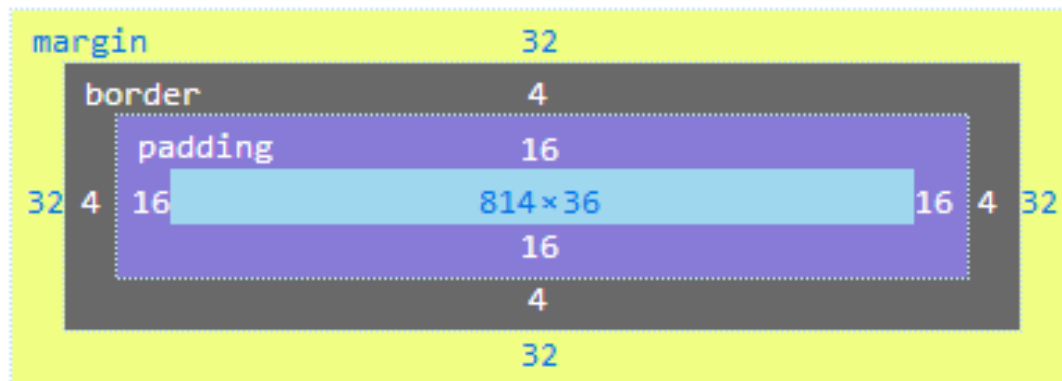
- On peut modifier les bordures de chaque côté avec
 - border-top
 - border-right
 - border-bottom
 - border-left
- Quelques exemples de border-style



Les marges

- Deux types de marges :

- Internes : padding
- Externes : margin



- Plusieurs façons de définir les marges/paddings :

```
margin : 1em;
```

Tous les côtés ont une marge de 1em

```
margin : 0.5em 1em;
```

Marges verticales de 0.5em ; marges horizontales de 1em


```
margin : 1em 2em 3em 4em;
```

Marge haute 1em ; marge droite 2em ;
Marge basse 3em ; marge gauche 4em



Cascade, héritage, spécificité

Cascade

- On peut appliquer des règles de style de plusieurs manières
- La « cascade » définit l'ordre de priorité des règles. Du plus prioritaire au moins prioritaire, on a :
 - Un style appliqué en CSS avec la valeur `!important` 
 - Le style appliqué directement sur l'élément (avec attribut `style`)
 - Le style appliqué en CSS
- Si plusieurs règles de même priorité sont en conflit, **la dernière règle déclarée l'emporte**

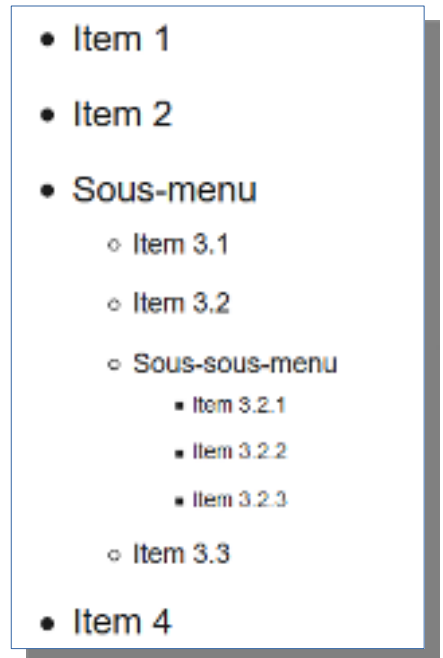
Héritage

- Certaines propriétés CSS sont transmises aux éléments enfant
On parle d'**héritage**

- Exemples :

- Vous fixez color et font-style à un élément.
Tous ses éléments enfant auront
la même couleur et style de police

- Vous réduisez la taille de police des listes
`ul { font-size:75% }`
Les listes imbriquées vont être aussi réduites



Spécificité

- Plus un sélecteur est **spécifique**, plus la règle CSS associée sera prioritaire

L'ordre de priorité est :

sélecteur d'**id** >>> sélecteur de **classe** >>> sélecteur de **type**

Exemple :

```
<p class="para" id="p2">Du faux-texte</p>
```

```
p {color : red ;}  
#p2 {color : green ;}  
.para {color : blue ;}
```

Du faux-texte

Spécificité

- Expliquée à la façon Star Wars

https://stuffandnonsense.co.uk/archives/css_specificity_wars.html

- Ou avec des poissons :

<https://www.specifishity.com/>

CSS is easy?

