

Technologies du Web

Master COMASIC

CSS

Antoine Amarilli¹

27 novembre 2014

1. Matériel de cours inspiré de notes par Pierre Senellart et de code par Pablo Rauzy. Merci à Pierre Senellart et Pablo Rauzy pour leur relecture.

Vue d'ensemble

- Cascading StyleSheets.
- Standard W3C :
 - CSS 1 1996
 - CSS 2 1998
 - CSS 2.1 2011, 487 pages
 - CSS 3.0 En cours (modulaire, support variable)
- CSS n'a pas une syntaxe XML (comparer à XML-FO).
- HTML décrit le **contenu** et la **structure**, CSS la **présentation**.
 - Avoir **plusieurs designs** pour un même site (www.csszengarden.com), en particulier pour **différents médias** (écran, impression, téléphone, lecteurs d'écran, robots...).
 - Utiliser le même design pour **plusieurs pages** (ou **sites**).

Table des matières

- 1 Introduction
- 2 CSS et HTML**
- 3 Généralités
- 4 Sélecteurs
- 5 Mise en forme
- 6 Mise en page
- 7 Autres fonctionnalités
 - Inclusion de contenu
 - Media queries

Intégrer avec HTML (solution 1)

- Couples **propriété/valeur**, par exemple :

```
color: red;
```

```
font-weight: bold;
```

- Peut s'appliquer **directement** aux éléments du document HTML avec l'attribut `style` :

C'est `<em style="color: red;">rouge`.

- Il faut qu'un élément existe ! Balises **neutres** `` et `<div>` (en-ligne et bloc).

Texte `en gras`.

→ **Problème** : mélange forme et contenu ; sur une page seulement.

Intégrer avec HTML (solution 2)

- **Sélecteurs** pour indiquer à qui s'applique une propriété.
- **Balise** `<style>` dans `<head>` :

```
<!DOCTYPE html>
<html>
  <head>
    <style type="text/css">
      em { color: red; }
    </style>
  </head>
  <body>
    Texte <em>en rouge</em>.
  </body>
</html>
```

→ Mêmes inconvénients.

Intégrer avec HTML (solution 3)

- Mettre les règles dans un fichier `.css` à part.
- Faire référence au fichier par la balise `<link>`.
- Possibilité de filtrer avec l'attribut `media` :

```
<!DOCTYPE html>
<html>
  <head>
    <link rel="stylesheet" type="text/css"
          href="style.css" media="screen">
  </head>
  <body>
    Texte <em>en rouge</em>.
  </body>
</html>
```

→ Séparation et réutilisabilité. (Mais dépendance externe.)

HTML et sélecteurs

- On va vouloir appliquer des règles à certains **éléments individuels** ou certaines **familles d'éléments**. Comment faire ?

→ Attributs `id` (**unique**) et `class` (**non-unique**).

```
<p id="remerciements">
```

```
  Merci à <span class="personne">Pierre</span> dont  
  j'ai utilisé les notes pour ce cours.
```

```
</p>
```

- Possible d'avoir **plusieurs classes** séparées par des espaces.

→ On pourra utiliser `id` et `class` dans CSS afin de sélectionner les bons éléments.

Table des matières

- 1 Introduction
- 2 CSS et HTML
- 3 Généralités**
- 4 Sélecteurs
- 5 Mise en forme
- 6 Mise en page
- 7 Autres fonctionnalités
 - Inclusion de contenu
 - Media queries

Syntaxe générale

```
/* Commentaires */
```

```
@charset "UTF-8";
```

```
@import url(autre.css);
```

```
@media screen {
```

```
    /* ... règles spécifiques ... */
```

```
}
```

```
selecteur1, selecteur2 {
```

```
    propriete: valeur;
```

```
}
```

Résolution des conflits (cascade)

1	!important	Les règles !important prévalent.
2	style	L'attribut style prévaut.
3	Spécificité	Le sélecteur le plus spécifique prévaut.
4	Ordre	La dernière règle prévaut.
5	Héritage	Si pas de règle, héritage des parents.
6	Défaut	Sinon, par défaut, par le navigateur.

- Le **comportement par défaut** diffère suivant le navigateur.
- **CSS reset** pour uniformiser le comportement des navigateurs.
- **Note** : Le @media n'a pas d'influence sur la spécificité.
- **Note** : Spécificité contre-intuitive ! #a versus #b c

Table des matières

- 1 Introduction
- 2 CSS et HTML
- 3 Généralités
- 4 Sélecteurs**
- 5 Mise en forme
- 6 Mise en page
- 7 Autres fonctionnalités
 - Inclusion de contenu
 - Media queries

Éléments, classes et IDs

```
strong {
  /* Mettre en rouge les choses importantes */
  color: red;
}

.personne {
  /* Souligner les noms de personne */
  text-decoration: underline;
}

p.important {
  /* Mettre en gras les paragraphes importants */
  font-weight: bold;
}

#logo {
  /* Rajouter une bordure au logo */
  border: 1px solid black;
}
```

Combinaisons

```
#menu li {
  /* Les li descendants de l'élément avec id="menu" */
}
header > img {
  /* Les img enfants d'un header */
}
h2 + p {
  /* Le premier paragraphe après un h2 */
  /* Aussi : ~ pour les successeurs. */
}
img[src*=".svg"] {
  /* Les images .svg */
}
ul > li:first-child {
  /* Le premier li immédiatement dans un ul */
}
```

Pseudo-éléments et pseudo-classes

```
p::first-line { /* Première ligne d'un paragraphe */ }  
h2::first-letter { /* Première lettre d'un titre */ }  
  
p::before { /* Avant un paragraphe */ }  
p::after { /* Après un paragraphe */ }  
  
a:link { /* Les liens qui sont des liens */ }  
a:visited { /* Les liens visités */ }  
a:focus { /* Les liens sélectionnés */ }  
a:hover { /* Les éléments survolés */ }  
a:active { /* Les liens cliqués */ }  
  
* { /* Sélecteur universel */ }
```

Table des matières

- 1 Introduction
- 2 CSS et HTML
- 3 Généralités
- 4 Sélecteurs
- 5 Mise en forme**
- 6 Mise en page
- 7 Autres fonctionnalités
 - Inclusion de contenu
 - Media queries

Unités de mesure

% Pourcentage de la **valeur** courante.

em Hauteur du bloc de la **police courante**.

px Taille absolue en **pixels**.

→ Dépend de la **résolution**.

cm Taille absolue en **centimètres**.

→ Surtout pour la version **imprimée**.

Texte

```
/* Taille de la police */  
font-size: 70%;  
/* Espacement des lignes */  
line-height: 150%;  
  
/* Espacement entre caractères */  
letter-spacing: 0.1em;  
/* Espacement entre mots */  
word-spacing: 1em;  
  
/* Gras */  
font-weight: bold;  
/* Italique (aussi : oblique) */  
font-style: italic;
```

Texte (suite)

```
/* Petites capitales */  
font-variant: small-caps;
```

```
/* Effets : underline, overline, blink, line-through */  
text-decoration: none;
```

```
/* Ombrage : décalage à droite, en bas, flou, couleur */  
text-shadow: 5px 5px 2px black;
```

```
/* Alignement dans un bloc : aussi left, right, center */  
text-align: justify;
```

Polices

- Les utilisateurs n'ont pas tous les mêmes **polices**.
- Liste de **priorités**, avec familles génériques :
⇒ serif, sans-serif, cursive, fantasy, monospace.

```
font-family: "DejaVu Sans Mono", monospace;
```

- Avec CSS3, possibilité de faire **télécharger** des polices :

```
@font-face {  
    font-family: maPolice;  
    src: url('maPolice.woff');  
}  
  
body {  
    font-family: maPolice;  
}
```

- **Web Open Font Format**, W3C, 2012. (Autres formats.)
- Une déclaration @font-face par variante (gras...).

Listes

```
ul {  
  /* Définir une image pour les puces de la liste */  
  /* Elle prévaudra sur list-style-type */  
  list-style-image: url('puce.png');  
  
  /* Choisir un style de puce prédéfini */  
  /* C'est utile si l'image de puce est inaccessible */  
  list-style-type: none;  
  /* Autres choix : disc, circle, square, decimal, */  
  /* lower-alpha, upper-alpha, lower-roman, upper-roman */  
}
```






Tableaux

`caption-side` Préciser si `<caption>` est en **haut** ou en **bas** (top/bottom).

`border-collapse` Fusionner les **bordures** des cellules (collapse ou par défaut `separate` qui est moche).
Si `separate`, utiliser `border-spacing` pour définir l'espacement.

`<col>` Permet de mettre un style de façon simple à toutes les cellules d'une **colonne**.
(Possible aussi avec `:nth-child(...)`.)

Couleurs et remplissages

Couleur	Nom CSS	RGB	Hexa
	black	rgb(0, 0, 0)	#000000
	blue	rgb(0, 0, 255)	#0000FF
	red	rgb(255, 0, 0)	#FF0000
	teal	rgb(0, 128, 128)	#008080
	-	rgb(32, 64, 96)	#204060

Autres remplissages possibles :

Images `url("image.jpg")`

Dégradés dépend des navigateurs

Utiliser couleurs et remplissages

```
/* Mettre le texte en blanc */  
color: white;  
/* Choisir un arrière-plan */  
/* Prévaut sur background-color */  
background-image: url("background.jpg");  
/* Ne pas faire défiler l'arrière-plan avec la page */  
background-attachment: fixed;  
/* Répéter l'arrière-plan verticalement et */  
/* horizontalement (réglage par défaut) */  
background-repeat: repeat;  
  
/* Mettre le fond en noir */  
background-color: black;
```

Table des matières

- 1 Introduction
- 2 CSS et HTML
- 3 Généralités
- 4 Sélecteurs
- 5 Mise en forme
- 6 Mise en page**
- 7 Autres fonctionnalités
 - Inclusion de contenu
 - Media queries

En-ligne et blocs

- Deux types d'**éléments** en HTML :
 - Les **blocs** : `<div>`, `<p>`, `<h1>`, ``...
 - Les éléments **en-ligne** : ``, `<a>`, ``, ``...
- On va parler du positionnement des **blocs**.
- Propriété CSS `display` pour contrôler le **mode d'affichage** :
 - none** Élément invisible.
 - Toujours accessible aux robots !
 - Comparer `visibility`: `hidden`.
 - block** Affiché comme un bloc.
 - inline** Affiché en ligne.
 - inline-block** Affiché en ligne, se comporte comme un bloc.
 - autres** Plein d'autres choix exotiques (`table-column-group`...).

Bordures

border-style Style de bordure :

- none
- solid (défaut)
- dotted, dashed, double
- groove, ridge, inset, outset (effets de relief, dépend de la couleur et nécessite de l'épaisseur)

border-color Couleur de la bordure.

border-width Épaisseur de la bordure.

border-collapse Utile pour les tables :
border-collapse: collapse.

border Raccourci :

```
border: 1px solid red;
```

```
border: 2px outset black;
```

Bordures (suite)

border-radius Arrondir la bordure. Possible de donner une valeur par coin, ou bordures elliptiques :

```
/* Arrondi de 10 px */  
border-radius: 10px;  
/* Arrondi NW, NE, SE, SW */  
border-radius: 0px 10px 10px 0px;  
/* Arrondis elliptiques */  
border-radius: 20px/10px;
```

box-shadow Ombrages sous un bloc :

```
/* Décalage à droite, en bas, flou, couleur */  
box-shadow: 5px 5px 1px black;  
/* Nécessaire pour la compatibilité */  
-webkit-box-shadow: 5px 5px 1px black;  
/* Idem, mais ombrage à l'intérieur */  
box-shadow: 5px 5px 1px black inset;
```

Largeur et hauteur

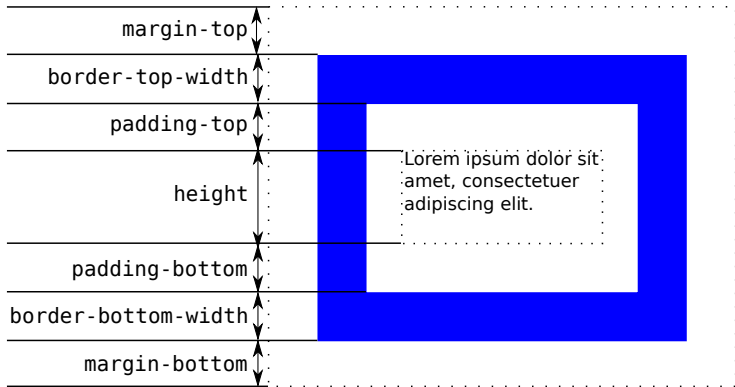
- `height` et `width` pour la **hauteur** et la **largeur**.
- Par défaut, `auto` (éléments aussi larges que **possible** et aussi hauts que **nécessaire**).
- Aussi pour les **images** `` bien que ce ne soit pas un **bloc**.
- `min-width` et `max-width` pour spécifier des **limites**.
 - Par exemple, pour imposer qu'un site ne soit pas **trop large**.

Débordement

- Si la taille est contrainte, le contenu peut **déborder**.
- Propriété `overflow` :
 - visible** Le contenu en trop est **visible** mais la taille de l'élément ne change pas pour **parents** et **voisins** (défaut).
 - hidden** Le contenu en trop est **invisible**.
 - scroll** Des barres de **défilement** sont ajoutées.
 - auto** Des barres de défilement sont ajoutées seulement **si nécessaire**.
- Aussi `overflow-x` et `overflow-y`.
- Les mots **trop longs** peuvent faire déborder la largeur :
 - `word-wrap`: `break-word` permet de **forcer la coupe**.

Marges et espacement

- Propriétés margin et padding.



- Cas particulier : `margin: auto` pour **centrer** un élément.
 - Il faut alors fixer `width`.
 - Ne pas confondre avec `text-align`.

Propriétés directionnelles

- Pour margin, padding, border-width... :
 - 1 valeur Tous les côtés.
 - 2 valeurs Haut et bas, gauche et droite.
 - 3 valeurs Haut, gauche et droite, bas.
 - 4 valeurs Haut, droite, bas, gauche.
- Aussi des propriétés **individuelles** :
 - margin-top, border-top-width...

Positionnement

- Propriété `position` : l'élément est positionné...
 - `static` ... par rapport au **flux** (défaut).
 - `relative` ... par rapport à sa **position standard** (décalage).
 - `absolute` ... par rapport à l'origine du **parent** le plus proche non `static` (à défaut, `<html>`).
 - `fixed` ... par rapport à la **fenêtre d'affichage**.
- Pour les trois derniers, utiliser `top bottom left right`.
- `z-index` définit l'**ordre vertical** en cas de chevauchement.
 - Ne fonctionne pas pour `static`.

Flottants et dégagement

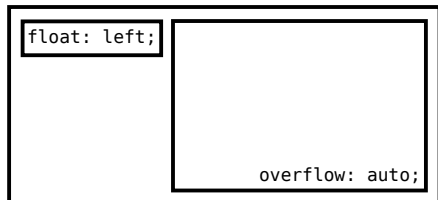
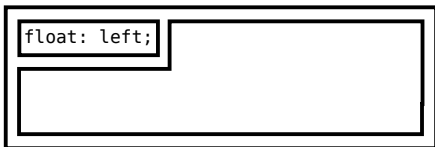
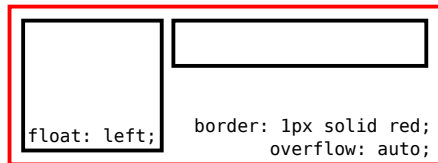
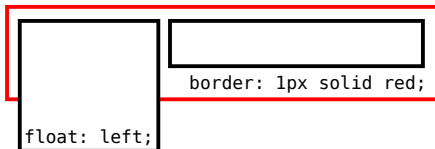
- float pour placer des éléments **hors du flux**.

```
div#menu {  
    float: right; /* aussi : left */  
}
```

- clear pour forcer la **resynchronisation**.

```
footer {  
    clear: both; /* aussi : left, right */  
}
```

Subtilités des flottants



Colonnes

- En CSS3, **colonnes** pour le texte :

```
p {  
  /* Largeur minimale et nombre maximal de colonnes */  
  columns: 300px 3;  
  /* Séparation entre les colonnes */  
  column-gap: 20px;  
  /* Trait entre les colonnes */  
  column-rule: 1px solid black;  
}
```

→ Préfixes spécifiques au moteur :

- webkit- pour Safari et Chrome.
- moz- pour Firefox.

Table des matières

- 1 Introduction
- 2 CSS et HTML
- 3 Généralités
- 4 Sélecteurs
- 5 Mise en forme
- 6 Mise en page
- 7 Autres fonctionnalités**
 - Inclusion de contenu
 - Media queries

:before et :after

- Pour insérer du texte **avant** ou **après** un élément.

```
p.reminder::before {  
  content: "N'oubliez pas :";  
}
```

- **Subtil** : quel texte relève de la **présentation** ? quel texte relève du **contenu** ?

Compteurs

Exemple pour **numéroter** sections et sous-sections.

```
article {
  counter-reset: section subsection;
}
article h2 {
  counter-increment: section;
  counter-reset: subsection;
}
article h2::before {
  content: counter(section) ". ";
}
article h3 {
  counter-increment: subsection;
}
article h3::before {
  content: counter(section) "." counter(subsection) ". ";
}
```

Media queries

- Spécifier des feuilles de style **différentes** suivant le **média**.
- **Attribut** media de <link> (permet d'éviter de charger les feuilles de style inutiles) ou **at-rule** @media en CSS.
- **Affichages** : screen, handheld (les smartphones, en théorie), tv, projection, tty, aural... et all.
- Test de la taille de la **surface de rendu** : max-width: 1024px (affichage de moins de 1024 px de largeur), etc.
- Test de la taille de l'**affichage** : device-width, etc.
- **Orientation** (smartphones) : orientation: portrait.

```
/* Version imprimable */
```

```
@media print
```

```
/* Malvoyants */
```

```
@media aural
```

```
/* Écrans de moins de 1024px */
```

```
@media screen and (max-width: 1024px)
```

Raffinements sur le mobile

- Attention, certains mobiles à **haute densité d'affichage** appliquent un facteur d'échelle par défaut entre les **pixels physiques** (petits) et les **pixels de CSS**.
 - **Différence** entre 100px en CSS et 100px physiques.
- Par défaut, les navigateurs mobiles font un rendu dans une **fenêtre virtuelle** (viewport) plus large que l'écran.
 - Les sites non prévus pour mobile sont affichés **correctement**.
 - L'utilisateur peut **zoomer** et se déplacer.
- Les sites prévus pour mobile (ou prévus pour s'afficher à faible largeur) peuvent forcer l'affichage à la largeur réelle :
`<meta name="viewport" content="width=device-width">`