Module 3

Style C.S.S

Cascading Style Sheets

Table des matières

STYLE CSS

Base CSS

CSS présentation

Cascading Style Sheets (CSS). Ce langage est venu compléter le HTML en 1996. Son rôle est de gérer l'apparence de la page web.

Tout comme HTML, le langage CSS évolue, nous en sommes aujourd'hui à la version 3 qui n'est supporté correctement que par les navigateurs modernes (à partir de Explorer9)

Exemple de structure HTML « classique » pour un site Web

```
<body>
      <div id="conteneur">
            <div id="entete"></div>
            <div id="menu"></div>
            <div id="page"></div>
            <div id="pied"></div>
      </div>
</body>
```

Suivant cet exemple, nous allons pouvoir dire en CSS : « je veux que le conteneur fasse 900px de large, que le menu se place à gauche de la page etc ... »
Toujours grâce au CSS, nous pourrons dire : « je veux que le texte soit de couleur bleue et tous les liens rouges sauf ceux contenus dans le menu qui seront verts » (et ce ne sera pas beau :-).

En reprenant l'exemple précédent, si je veux que **le conteneur du site ait une largeur de 900px et une couleur de fond noire**, la syntaxe sera la suivante

Exemple de code CSS

```
#conteneur {
      width: 900px;
      background-color: #000000;
}
```

Lier CSS et HTML

3 façons différentes, **la 3ième** est à privilégier :

1- Introduire le CSS dans la balise head de votre fichier HTML

```
<head>
      <style type="text/css">
      #conteneur {
      width: 900px;
      background-color: #000000;
      }
      </style>
</head>
```

Cette méthode peut être pratique lorsqu'on a une **page unique**, mais doit être évitée si il y a plusieurs pages.

2- Styliser une balise directement

Ainsi, dans notre exemple, à la place de

```
<div id="conteneur"> ... </div>
```

nous aurions :

```
<div style="width: 900px; background-color: #000000;"> </div>
```

Cette méthode **est à éviter** car difficilement maintenable, mais peut être employée pour forcer un style.

3- Lier un fichier externe .css

Cette **méthode est à retenir** car ainsi le code est :

- Plus clair : distinction marquée entre CSS et HTML.
- Plus facile à modifier, à faire évoluer : **un même fichier CSS peut être lié à 100 pages HTML**, il suffira de faire une seule modification dans le fichier .css pour qu'elle s'applique aux 100 pages.
- Plus rapide à mettre en place : certains fichiers .css peuvent être réutilisés d'un site à l'autre.

Vous pouvez nommer ce fichier comme vous voulez. Par convention ce fichier s'appelle style.css. Vous pouvez par exemple le préfixer du nom du site auquel il est rattaché pour plus de lisibilité : mon_site_style.css

Dans la partie **<head></head>** de votre site pour lier un fichier css externe :

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

Attributs

- `rel` définit la nature de la relation établie entre deux ressources.
- `type` : définit la nature du fichier inclus
- `media` : peut prendre ici les valeurs :
  - «`all`» : tous types de media,
  - « `print` » : pour une feuille de style destinée à l'impression ou les couleurs seront supprimées par exemple,
  - « `screen` » : affichage à l'écran
  - « `projection` » : video projecteur
  - « `tv` » : television
  - « `aural` » : synthetiseurs vocaux
- **href** : permet de faire le lien avec la feuille de style depuis la page HTML. C'est cette valeur que nous allons faire varier.

Par exemple, pour lier une feuille de style nommée **blog_style.css** se trouvant dans le sous-dossier [**css**] :

```
<link rel="stylesheet" href="css/blog_style.css" type="text/css"
media="all" />
```

Appliquer un style

Pour appliquer un style à un élément HTML : p, div, strong... La syntaxe CSS est toujours la même :

```
balise1
{
    propriete1 : valeur1 ;
}

balise2
{
    propriete1: valeur1;
    propriete2: valeur2;
    propriete3: valeur3;
    propriete4: valeur4;
}
```

Le style peut être appliqué à un type de balise, en l'appelant par **son nom** par exemple : **p**, c'est à dire à **tous** les paragraphes.

Une balise peut être identifiée grâce à un **attribut** HTML `id` ou `class` dont la valeur est libre. Cela permettra de **cibler plus précisément** certains éléments HTML pour leur appliquer un style.

Exemple :

HTML
```
<p>Ici mon texte apparaîtra en noir</p>

<p>Ici mon texte apparaîtra en noir mais je peux décider qu'un mot sera
écrit en <span class="bleu">bleu</span>.
Comme l'attribut utilisé est une classe je peux le réutiliser pour
définir <span class="bleu">un autre portion de texte en bleu</span></p>

<p class="rouge">Ici mon texte apparaîtra en rouge</p>
```

CSS
```
p{
  color: black ;
}
.rouge{
  color: red;
}
.bleu {
  color : blue;
}
```

Remarque : la balise **span** est une balise inline qui va permettre d'appliquer un style particulier à un endroit du texte.

Attribut class ou id ?

Ces 2 attributs ont exactement la même fonction : identifier (coller une étiquette à) un élément HTML pour lui appliquer un style particulier.

Différences

1- La première différence entre ces 2 balises est que l'attribut class peut être réutilisé autant de fois que l'on veut dans une page (`<span class="bleu">...</span>`) alors que **la valeur d'un attribut id ne sera utilisée qu'une seule fois dans la page** (`<div id="conteneur">...</div>`) : un seul conteneur principal.

Mauvais code :
```
<body>
      <ul id="menu">
            <li id="liste_menu"> ...</li>
            <li id="liste_menu"> ...</li>
            <li id="liste_menu"> ...</li>
      </ul>
</body>
```

Bon code :

```
<body>
     <ul id="menu">
          <li class="liste_menu"> ...</li>
          <li class="liste_menu"> ...</li>
          <li class="liste_menu"> ...</li>
     </ul>
</body>
```

2- La seconde différence est que l'attribut `class` sera « appelé » en CSS avec un point :

- `liste_menu {}` alors que l'id sera appelé en CSS avec un dièse : **#**`menu{}`

3- La troisième différence est que l'attribut **id** a **plus de « poids »** en cas de conflit que l'attribut `class` pour déterminer un style particulier « cf. chapitre 5.6 Poids des sélecteurs».
Pour comprendre ce concept, il faut étudier la notion de « cascade » dans les feuilles de style, ce qui implique de comprendre la syntaxe des sélecteurs.

Utilisation

Vous pouvez vous dire : « pour ne pas se tromper, il suffit de mettre des class partout », ce qui n'est pas faux. Cependant, habituellement, les éléments uniques d'une page sont identifiés par un id, pour plus de lisibilité dans le code. Si vous voyez un dièse en CSS (=id en HTML) vous pouvez être sûr qu'il s'agit d'une zone unique.

Ainsi, la plupart du temps, les éléments structurants d'un site sont définis par des id :

```
<div id="conteneur">...</div>, <div id="entete"></div>,
<div id="pied"></div> …
```

Dans la pratique les intégrateur utilisent essentiellement l'attribut **class**

A noter :

1- Il est tout à fait possible (souvent utilisé) d'**appliquer plusieurs classes à un même élément**. Elles seront **séparées par un espace**.

HTML

```
<body>
     <p class="red intro">Ici mon texte d'introduction</p>
</body>
```

CSS

```
body{
	font-size : 12px;
	color:#000000;
}
.red{
	color : #850d0d;
}

.intro {
	font-size : 16px ;
}
```

2- La valeur de l'attribut **class ou id ne peut pas commencer par un chiffre**, contenir des caractères spéciaux, accents ou espaces.

TP8_lier_css

Les sélecteurs

Sélecteur universel

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

Ce symbole s'applique à **tous les éléments** de la page (à utiliser avec précautions). Il est souvent utilisé pour « mettre les compteurs à zéro », ici les marges intérieures et extérieures doivent être à zéro par défaut.

Appliquer un style à une balise

```
p {
	font-size:12px;
}
```

Tous les textes contenus dans des paragraphes auront une taille de 12 pixels.

Identifier une balise par son attribut class

```
img.presentation {
	border : 1px solid red;
}
```

Toutes les images ayant l'attribut `class="presentation"` auront une bordure rouge (les autres non)

```
.presentation {
      border : 1px solid red;
}
```

Ici la bordure pourra être appliquée à différents éléments HTML et pas simplement aux images.
Privilégier cette syntaxe plutôt que la première pour plus de souplesse.

Identifier une balise par son attribut id

```
#page {
      background-color : black;
}
```

L'élément **(unique)** ayant l'attribut `id="page"` aura une couleur de fond noire.

Un même style pour différentes balises

```
p.introduction, blockquote {
      font-weight : bold;
}
```

Tous le paragraphes ayant l'attribut class="introduction" **ET** toutes les citations seront en gras.

Préciser une balise contenues dans une autre

```
.introduction a {
      text-decoration: underline;
}
```

Tous les liens contenus **DANS** le(s) éléments ayant l'attribut class="introduction"

Préciser une balise qui suit une autre

```
h1+p {
      font-style:italic;
}
```

Le paragraphe situé directement **APRES** un titre de niveau 1 sera en italique

Cibler seulement le descendant direct d'un élément

```
#menu > li > a {
	background-color: #000000;
}
```

```
<ul id="menu">
	<li><a href="#">Lien</a></li>
	<li><a href="#">Lien</a></li>
	<li><a href="#">Lien</a>
		<ul>
			<li><a href="#">Lien niveau 2</a></li>
			<li><a href="#">Lien niveau 2</a></li>
		</ul>
	</li>
</ul>
```

Tous les éléments `a` **qui sont les descendant directs** d'un élément `li` lui même directement contenu dans un élément ayant l'identifiant menu.

Cibler les éléments en fonction de leur attribut

```
a[title]{
	border-bottom: 1px dotted #cccccc;
}
```

Tous les liens qui ont un tilte (infobulle) seront soulignés en pointillés gris

```
input[type="text"]{
	border-bottom: 1px solid #cccccc;
}
```

Tous les champs (zone de saisie d'un formulaire) de type text (pas les types email, tel,,,)

Voir une lise des sélecteurs : http://www.w3schools.com/cssref/css_selectors.asp

TP9_menu

Pseudos classes

*"Une **pseudo-classe** CSS est un mot-clé ajouté au sélecteur pour indiquer un état particulier de l'élément qui doit être sélectionné. Par exemple, `:hover`, appliquera le style quand l'utilisateur survolera l'élément visé par le sélecteur." https://developer.mozilla.org/fr/docs/Web/CSS/Pseudo-classes*

Liens

```
a{
	color: #000000;
}

a:hover {
	background-color: #000000;  /* Couleur de fond au survol */
	color: #ffffff;
}
```

Tous les liens auront un fond noir et une couleur de police blanche au survol.

Également pour les liens, pseudos classes :

- :active – lien qui est activé : quand l'internaute clique dessus
- :focus - élément sélectionné (ex:champ de formulaire)
- :link - liens qui n'ont pas été visités (peu utilisé)
- :visited – tous les liens qui ont été visités (peu utilisé)

En savoir plus : http://www.alsacreations.com/astuce/lire/43-comment-dfinir-lapparence-de-ses-liens.html

Autres pseudo-classes

Il existe beaucoup de pseudo-classes, voici les plus utilisées :

- `:first-letter` - Sélectionne la première lettre d'un élément
- `:first-line` – Première ligne
- `:first-child` – Sélectionne le premier élément enfant
  (également `:last-child` et `:nth-child(5)` pour le 5ième élément)

Formater la première lettre de tous les titres de niveau 2 :

```
h2:first-letter {
	font-size : 30px;
	color: #ccc; }
```

- `:before` - Insertion avant l'élément

- `:after` - Insertion après l'élément

Souvent utilisé avec la propriété *content* pour insérer du texte (ex : h1:before{content:"chapitre :"}) ou une image a:after{content:url(img/mon-image.png)}

- :not(p) - Qui n'est pas un p

- :target - applique un style à l'élément ciblé via une ancre
  Exemple d'onglets en css : http://www.w3schools.com/cssref/tryit.asp?filename=trycss3_target_tab

TP autonome :

+ ciblez tous les liens qui ont l'attribut `target="_blank"` et liez leur l'image (avant ou après) `blank.png` (elle de trouve dans [documents])

- W3C

+ Faites une liste et entourez d'une bordure le 2ieme élément

- element 1
- element 2
- element 3
- element 4

Poids des sélecteurs

Une notion importante concernant les feuilles de style est celle de cascade. **Les éléments enfants héritent des caractéristiques de leurs parents**.

2 styles contradictoires peuvent être définis pour un même élément (ex : titre en rouge ET titre en bleu) le style s'impose pour le sélecteur qui a le plus de poids

HTML

```
<body>
  <div id="colonne_droite">
     <p>Ici mon texte </p>
  </div>
  <div id="page">
     <p>Ici mon texte</p>
     <p class="special">Un texte un peu spécial</p>
  </div>
</body>
```

CSS

```
body {
     font-family:Arial;
     /* Tout le contenu sera en Arial */
}


#page p {
     font-family:Georgia;
     /* La page sera en Georgia et la colonne droite reste en Arial */
}

#page .special {
     font-family:Helvetica;
}
```

Le dernier sélecteur (#page .special) étant suffisamment « lourd » (111) il prendra le pas sur #page p (101) et la police choisie s'appliquera.

Par contre, si on avait seulement écrit : .special (11) c'est la police de #page : Georgia qui se serait imposée.

Voici le poids de chaque type de sélecteur

- *** {...}** : 0000 (aucun identifiant, aucune classe, aucun élément) ;
- **p {...}** : 0001 (aucun identifiant, aucune classe, un élément) ;
- **blockquote p {...}** : 0002 (aucun identifiant, aucune classe, deux éléments) ;
- **.class {...}** : 0010 (aucun identifiant, une classe, aucun élément) ;
- **p.class {...}** : 0011 (aucun identifiant, une classe, un élément) ;
- **blockquote p.class {...}** : 0012 (aucun identifiant, une classe, deux éléments) ;
- **#id {...}** : 0100 (un identifiant, aucune classe, aucun élément) ;
- **p#id {...}** : 0101 (un identifiant, aucune classe, un élément) ;
- **blockquote p#id {...}** : 0102 (un identifiant, aucune classe, deux éléments) ;
- **.class #id {...}** : 0110 (un identifiant, une classe, aucun élément) ;
- **.class p#id {...}** : 0111 (un identifiant, une classe, un élément) ;
- **blockquote.class p#id {...}** : 0112 (un idenfifiant, une classe, deux éléments) ;
- **<p style="...">** : 1000 (attribut HTML style qui ne sera supplanté que par un style utilisateur normal) ;

Source : open Web

Formatage du texte

Taille du texte

```
h1 {
      font-size: 16px; /* equivaut à  1em */
}
```

Les unités de mesures en CSS (pour exprimer des largeurs, marges, taille de police...) :

- en px (absolu),
- pt (absolu),
- em (relatif),
- % (relatif)
- rem (relatif-absolu)

Pts	Px	Em	Porcentaje
6pt	8px	0.5em	50%
7pt	9px	0.55em	55%
7.5pt	10px	0.625em	62.5%
8pt	11px	0.7em	70%
9pt	12px	0.75em	75%
10pt	13px	0.8em	80%
10.5pt	14px	0.875em	87.5%
11pt	15px	0.95em	95%
12pt	**16px**	**1em**	**100%**
13pt	17px	1.05em	105%
13.5pt	18px	1.125em	112.5%
14pt	19px	1.2em	120%
14.5pt	20px	1.25em	125%
15pt	21px	1.3em	130%
16pt	22px	1.4em	140%
17pt	23px	1.45em	145%
18pt	24px	1.5em	150%
20pt	26px	1.6em	160%
22pt	29px	1.8em	180%
24pt	32px	2em	200%
26pt	35px	2.2em	220%
27pt	36px	2.25em	225%
28pt	37px	2.3em	230%
29pt	38px	2.35em	235%
30pt	40px	2.45em	245%
32pt	42px	2.55em	255%
34pt	45px	2.75em	275%
36pt	48px	3em	300%

Précisions sur "em" :

"*En écrivant* `font-size: Xem` *on ne demande pas une taille de texte fixe et absolue, mais une taille de texte proportionnelle à la taille de texte de l'élément parent.* "

Voir : http://www.alsacreations.com/article/lire/563-gerer-la-taille-du-texte-avec-les-em.html#renvoi2

IL faut "initialiser" en indiquant la taille du texte par défaut en indiquant, lorsqu'on utilise des em :

```
html{
 font-size: 100%; /* Pour les vieux navigateurs */
}
body{
 font-size:0.95em; /* Choisir la taille de police par défaut */}
```

Assez complexe à employer, le plus simple à utiliser reste le px mais le rem offre des possibilités intéressantes : http://www.pompage.net/traduction/dimensionner-ses-fontes-avec-rem

Police

```
p { font-family: Arial, "Arial Black", Verdana, sans-serif; }
```

Pour qu'une police s'affiche correctement, il faut que tous les internautes l'aient. Si un internaute n'a pas la même police que vous, son navigateur prendra une police par défaut (une police standard).

Pour cette raison, il est fortement conseillé d'inscrire **plusieurs noms de polices séparées par des virgules**.

Le navigateur essaiera d'abord d'utiliser la police1. S'il ne l'a pas, il essaiera la police2. S'il ne l'a pas, il passera à la police3, et ainsi de suite.

Si le nom de la police comporte des espaces il faut l'entourer de guillemets ex ''Arial Black''

Voici une liste de polices qui fonctionnent bien sur la plupart des navigateurs :

- Arial ;
- Arial Black ;
- Comic Sans MS ;
- Courier New ;
- Georgia ;
- Impact ;
- Times New Roman ;
- Trebuchet MS ;
- Verdana.

TP Guidé : Insérer une police hébergée sur Google Fonts : https://www.google.com/fonts

Italique, gras, etc

Italique

```
h3{
	font-style: italic; /* ou normal */
}
```

Gras

```
p.intro{
	font-weight: bold;
}
```

Valeurs possibles :

- **bold**
- **bolder**
- **normal**

Souligné

```
p.important, a:hover {
	text-decoration: underline;
}
```

Pour annuler un soulignement (liens) : text-decoration: none;

Barré

```
p.nul {
	text-decoration: line-through;
}
```

Coloré

```
a.noir {
      color: black;
}
```

Valeurs possibles :

- **Code couleur hexadécimal de type #000000**
- **Code rgb : rgb(53,0,8)**

Hauteur de ligne

```
p{
      line-height:16px;
}
```

Alignement du texte

```
h1{
      text-align: center;
}
```

Valeurs possibles :

- **left**
- **right**
- **justify**
- **center**

Note : l'alignement ne peut se faire que sur des éléments de type block (`<h1>` `<p>` ...)

Retrait de texte (première ligne)

```
.intro{
      text-indent:20px;
}
```

En majuscule

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

Espace entre 2 mots

```
h1{
	word-spacing: 4px;

}
```

Espace entre 2 lettres

```
h1{
	letter-spacing: 1px;

}
```

Césures

Pour éviter les dépassements d'un bloc dû à un mot trop long dans un conteneur trop étroit, utiliser la propriété suivante, appliquée au conteneur.

```
.colonne-droite{
	word-wrap: break-word; /* Au besoin, "casser" le mot pour éviter qu'il ne dépasse */
}
```

Voir l'article : http://www.alsacreations.com/tuto/lire/1038-gerer-debordement-contenu-et-cesures-css.html

TP10_style_texte

Positionnement des éléments

Je vais donc déterminer en HTML des « boites » (toutes les balises) que je vais positionner et « habiller » grâce au CSS. Ceci constituera la structure de mon site Web.

Largeur et hauteur

Il est important de garder à l'esprit que si la largeur d'un site est en général fixée, la hauteur de chaque élément est variable, le texte librement administrable pouvant s'allonger (page et menu notamment).

Les propriétés width et height permettent de fixer les dimensions d'une div ou tout autre élément de type block

Exemple :

HTML

```
<body>
      <div id="boite"></div>
</body>
```

CSS

```
#boite {
      width: 200px ; /* largeur */
      height : 200px ; /* hauteur */
      border : 1px solid #000 ; /* sinon je ne vois rien ! */
}
```

A noter : la largeur d'un élément peut être fixée en % ce qui permet d'avoir une structure fluide. Valeur par défaut : auto (automatique)

Existent également les propriétés :

- `min-width` et `max-width` = largeur minimum et maximum autorisées.
- **`min-height`** et `max-height` = hauteurs minimum et maximum autorisées.

Note : **Il vaut mieux utiliser min-height plutôt que height**, ce qui permet à l'élément de s'allonger si son contenu le "pousse".

Exemple :

CSS

```
#boite {
     width: 90 %;
/* occupe 90 % de la balise body : visible au redimensionnement de la
     fenêtre*/

     min-width: 400px;
/* mais même si on réduit la fenêtre du navigateur à 200px de large, la
largeur de la boite se fixera à 400px;

     max-width: 1000px;
/* De même, sur un écran large à + de 1300px, la taille de la boite ne
dépassera pas 1000px */

     min-height:300px;
/* La hauteur de la boite est de 300px minimum mais si du contenu la fait
s'allonger, elle pourra avoir une hauteur plus important. Remarque : la
propriété max-height est rarement utilisée */
}
```

Les éléments flottants

Nous avons vu que par défaut, les étant des éléments de type block, elles se positionnent les unes au dessous des autres. Hors si je veux placer mon menu à gauche de ma page par exemple, il va me falloir utiliser la propriété CSS : `float`.

Je vais faire flotter mon menu vers la gauche en lui appliquant `float:left` ou à droite `float:right` annuler l'effet d'un flottant- `float:none`

Exemple :

HTML

```
<div id="conteneur">
  <div id="menu">
    <h3>Menu</h3>
  </div>
  <div id="page">
    <h1>Page</h1>
  </div>
</div>
```

CSS
```
#conteneur, #menu, #page{
	border:1px solid #000000;
}

#conteneur{
	width: 900px;
	margin:auto;

}

#menu{
	width: 250px;
	min-height: 200px;
	float:left;

}
#page{
	width: 600px;
	min-height: 400px;
	float:right;
}
```

Propriété clear

Les éléments flottants acceptant que les balises qui les suivent viennent se placer à coté, pour **placer un élément SOUS un autre ayant la propriété float**, il va falloir donner à cet élément la **propriété clear**.

HTML
```
<div id="container'>
  <div class="float"></div>
  <div class="float"></div>
  <div id="footer"></div>
</div>
```

CSS
```
.float{
	float:left;
	width: 200px;
	min-height: 200px;
}

#footer {
	clear : both;
```

```
}
/* Permet de « nettoyer » après un élément flottant valeurs possibles :
left (après flottant gauche), both (les deux), right (après flottant
droite) */
```

Dans cet exemple, le footer restera bien sous les autres éléments. De plus le container s'adaptera en hauteur aux éléments qu'il contient.

Hack clearfix

Pour éviter que les éléments flottants ne *« débordent »* de leurs conteneurs il est possible d'appliquer au conteneur une propriété clear appliquée avec la pseudo class *:after* afin de gérer au mieux la mise en page. Ce bout de code est associé à une classe appelée le plus souvent *.clearfix*

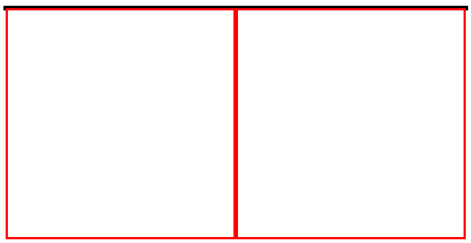
```
.clearfix:before,
.clearfix:after {
    content: " ";
    display: table;
}

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

Sans clearfix :

```
<div id="container"></div>
  <div class="float"></div>
  <div class="float"></div>
</div>
```

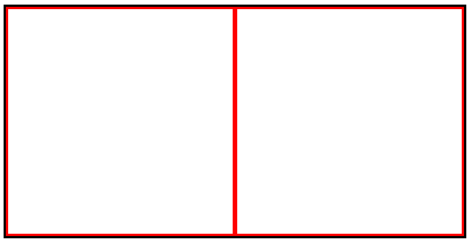
Le conteneur (bordure noire) se referme sur les 2 éléments flottants



Avec clearfix :

```
<div id="container" class="clearfix"></div>
  <div class="float"></div>
  <div class="float"></div>
</div>
```

Le conteneur entoure bien les éléments flottants



TP11_gabarit

Propriété Display

Il est possible de changer le type de comportement "inline" ou "block" par défaut des balises HTML grâce à la propriété CSS "display"

Valeur "inline" ou "block"

Nous avons vu que les éléments de type block se placent par défaut les uns au dessous des autres et occupent toute la largeur du conteneur dans lequel ils sont placés. Contrairement aux éléments de type inline qui se comportent comme des flottants.

Ainsi, un titre (h1) qui est un élément de type block peut réagir comme un élément de type inline si je lui applique la propriété CSS :

```
h1 {
      display : inline;
}
```

Également, si je veux par exemple qu'un lien occupe toute la largeur d'un conteneur pour avoir une plus grande zone cliquable, je définirai en CSS :

```
a{
      display : block;
}
```

Valeur "inline-block"

Les éléments de rendu **inline-block** conservent les mêmes caractéristiques que les "inline", mais peuvent être dimensionnés, par exemple la balise `<input />`.

Valeur "table-cell"

Article sur Alsacréations des possibilités de cette valeur (centrage vertical) :
http://www.alsacreations.com/tuto/lire/610-Mise-en-page-CSS-avancee-grace-a-la-propriete-display.html

Valeur "none"

Enfin la propriété `display:none`; fait « disparaître » l'élément concerné. Le code HTML reste mais l'élément n'apparaîtra pas à l'écran.

Toutes les valeurs possibles pour la propriété display :
http://www.w3schools.com/cssref/pr_class_display.asp

Valeur "flex"

Nous verrons dans le prochain module les possibilités offertes par cette nouvelle propriété permettant de faire des mises en pages avancées.

Types de positionnements (position)

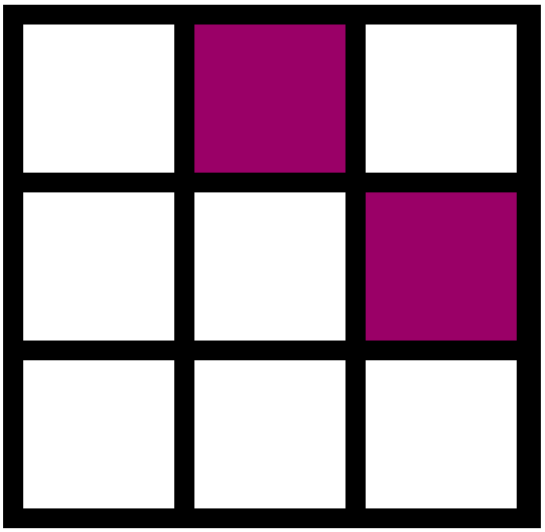
position : relative

Dans la majorité des cas il vaut mieux favoriser le positionnement relatif, c'est à dire que les éléments se placent les uns par rapport aux autres (en dessous ou à côté). A privilégier notamment lorsque l'on veut créer un site à largeur variable.

C'est le type de positionnement que nous utilisons depuis le début. Lorsque rien n'est précisé, les navigateurs appliquent « naturellement » la propriété : `position:relative` à tous les éléments HTML.

Les éléments sont positionnés en utilisant la propriété CSS margin (margin-left : 25px; Pour déplacer le bloc vers la droite). En les faisant flotter ou non.

TP autonome
Reproduisez ce modèle dans un nouveau fichier "portfolio.html" > "portfolio_style.css"
Carrés de 150px de côté, marges de 20px. Utilisez des balises `<div>` et l'attribut `class`. Couleur de fond noire (pas de bordure)



position : absolute

Un élément en position absolue peut être placé n'importe ou dans la page **en référence à son premier élément parent positionné** (=élément conteneur ayant lui même la propriété CSS position : relative **ou** position : absolute) Ou, à défaut d'élément positionné, de la fenêtre du navigateur.

L'élément sera placé grâce aux propriétés : right, left, top, bottom (ex : left: 25px; top:10px;)

Utiliser le positionnement absolu pour (non exhaustif) :

- **placer une boite (div) au-dessus d'une ou de plusieurs autres (à la façon d'un calque dans photoshop).**
- **changer l'ordre des éléments de la page (mettre en premier les textes importants) pour gagner en accessibilité et référencement.**
- **« attacher » un élément en bas d'un conteneur (ex:bloc3)**

Exemple :

HTML

```
<body>
<div id="conteneur">

  <div id="bloc1">
    <h2>Bloc 1</h2>
    <p>Position:absolute; z-index:1</p>
  </div>

  <div id="bloc2">
    <h2>Bloc 2</h2>
    <p>Position:absolute; z-index:2</p>
  </div>

  <div id="bloc3">
      <h2>Bloc 3</h2>
      <p>Position:absolute; z-index:3</p>
  </div>

</div>
</body>
```

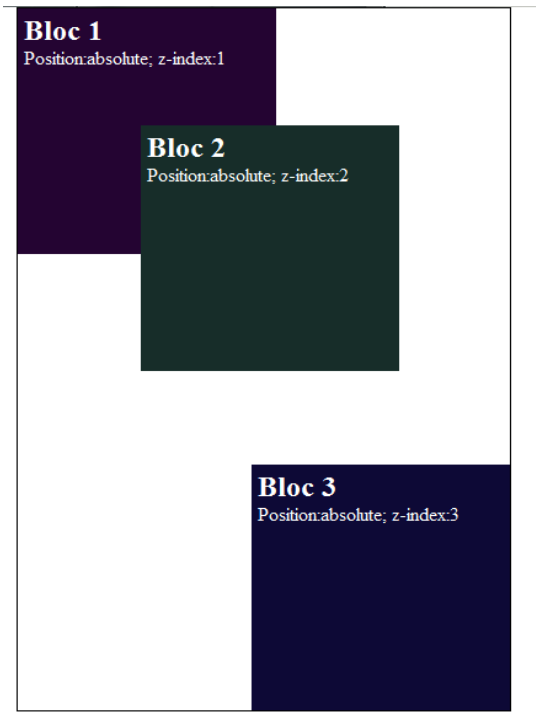
CSS positionnement absolu

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

#bloc1, #bloc2, #bloc3{
      position: absolute; /* Déclaration du type de positionnement */
      width:200px;
      height:200px;
      padding:5px;
}

#conteneur{
      width: 400px; /* Pour etre positionné,l'élément parent doit avoir
```

```
une largeur fixée  */
      height:600px;
      margin:auto;
      border:1px solid #000000;
      position: relative; /* Important de préciser le type de
positionnement de l'élément parent  */
      color: #FFFFFF;
}
#bloc1{
      background-color: rgb(35,5,49);
      z-index:1; /* Plus ce chiffre est élevé plus l'élément sera "au
dessus" */
      top:0;
      left:0;
}
#bloc2{
      background-color: rgb(22,44,40);
      z-index:2;
      top:100px;
      left:100px;
}
#bloc3{
      background-color: rgb(13,10,53);
      z-index:3;
      bottom:0;
      right:0;
}
```



position : fixed

Le positionnement fixe s'apparente au positionnement absolu, à l'exception des points suivants:

- Lorsque le positionnement est précisé (`top`, `right`, ...), l'élément est *toujours* positionné par rapport à la **fenêtre du navigateur**
- L'élément est fixé à un endroit et ne pourra se mouvoir, même lors de la présence d'une barre de défilement. En d'autres termes, la position initiale est fixée au chargement de la page, le fait qu'une éventuelle scrollbar puisse être utilisée n'a aucune influence sur le positionnement de l'élément: il ne bouge plus de la position initialement définie.

Source : Alsacreations

TP guidé

Faire des tests de positionnement en absolu, relatif, fixed dans le fichier tests.html

TP12_blog1

Habillage des éléments

Plusieurs propriétés CSS vont permettre « d'habiller » les div et autres éléments ainsi positionnées

Couleur de fond

La propriété background-color pour donner une couleur de fond à un élément. Quel que soit le type d'élément ! Vous pouvez donner une couleur de fond à un titre, une div, une liste...

La valeur peut être hexadécimale : #214487 ou bien exprimée en rgb

CSS - exemple :

```
h1{
background-color : #214487;
/*  background-color : rgb(13,27,43); */
}
```

Avec CSS3 il est possible de rendre la couleur de fond transparente

La notation RGBa obéit aux mêmes règles de fonctionnement que la notation classique RGB, mis à part qu'une composante est ajoutée à la valeur : `rgb(0,0,0)` devient donc `rgba(0,0,0,0.2)`. La dernière valeur (l'alpha) indiquant le degré d'opacité entre 0 et 1.

```
h1{
 background-color : rgba(13,27,43,0.5);
}
```

Malheureusement cette notation n'est pas comprise par ie7 et 8. Pour l'imposer à ces navigateurs anciens, il existe des hacks complexes ou simplement utiliser une image de fond .png transparente. (Cf dégradation élégante : http://www.pompage.net/traduction/degradation-elegante-et-amelioration-progressive)

Image de fond

Pour lier une image de fond, la propriété : `background-image` la valeur donnée est celle de l'URL liant votre fichier CSS à l'image. Différentes propriétés viennent compléter `background-image :`

background-repeat : répétition du fond

- **`no-repeat` : le fond ne sera pas répété. L'image sera donc unique sur la page.**
- **`repeat-x` : le fond sera répété uniquement sur la première ligne, horizontalement.**
- **`repeat-y` : le fond sera répété uniquement sur la première colonne, verticalement.**
- **`repeat` : le fond sera répété en mosaïque (par défaut).**

background-position : position du fond (pour un fond ne se répétant pas)

La valeur peut être donnée en pixel par rapport au coin supérieur gauche de l'élément.

Exemple : `background-position : 10px 25px;`
>Image placée à 10px du bord gauche et 25px du haut

De même : `background-position : top left;`
> Image placée en haut à gauche (valeurs possibles : `top, bottom, left, right, center`)

background-attachment

Par défaut le fond défile lorsqu'il y un scroll dans la page. Pour garder le fond fixe, appliquer la valeur "fixed".

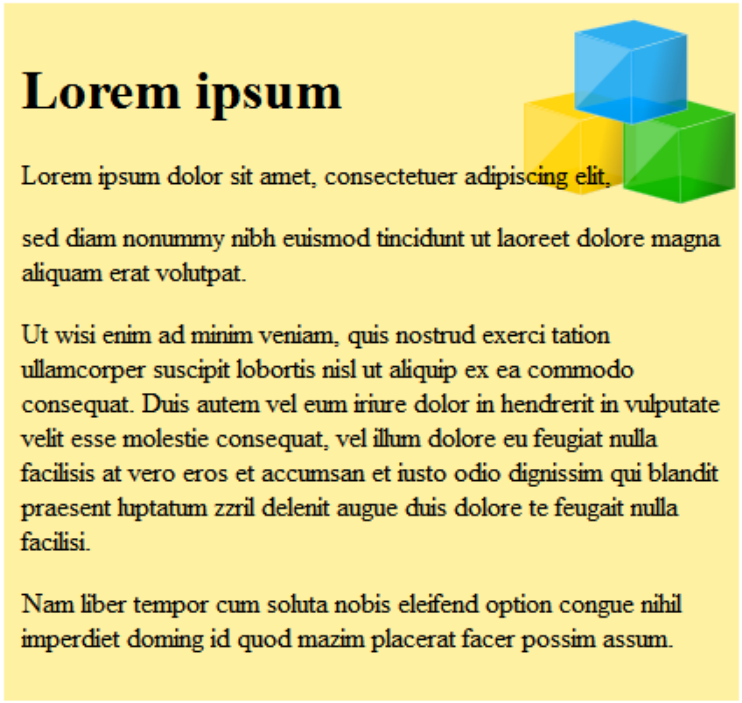
Exemple

CSS - définir une image ET une couleur de fond

```
#conteneur{
	background-image:url("img/cubes.png");
	background-position:right top;
	background-repeat: no-repeat;
	background-color: #fff0a3;
	background-attachment : fixed;
}
```

Ce code peut être « synthétisé » de la manière suivante

```
#conteneur {
	background: url("img/cubes.png") right top no-repeat #fff0a3 fixed;
}
```



background-size

Régler la taille de l'image de fond. Intéressant pour les largeurs fluides avec la valeur **`cover`** afin que l'image de fond "couvre toujours entièrement son conteneur.
Exemples : http://www.w3schools.com/cssref/playit.asp?filename=playcss_background-size&preval=cover

Balise img ou propriété CSS background ?

Une image insérée en background n'est pas détectée par les navigateurs, dans la création d'un site, il s'agit d'une sorte de « papier peint » : éléments de décoration, fond de bouton, de titre ...

Si l'image introduite est un élément **apportant une information** : logo, photo, carte ... Elle doit être insérée avec la balise **** : c'est préférable pour des questions d'accessibilité et de référencement S.E.O.

TP13_background

Bordures

De nombreuses propriétés CSS permettent de modifier l'apparence des bordures. Là encore tous les éléments et pas seulement les div peuvent bénéficier d'une bordure (li, ul, h1 ...)

Exemple :

CSS

```
border-width: 2px;
border-style : solid;
border-color : #fff0a3;
```

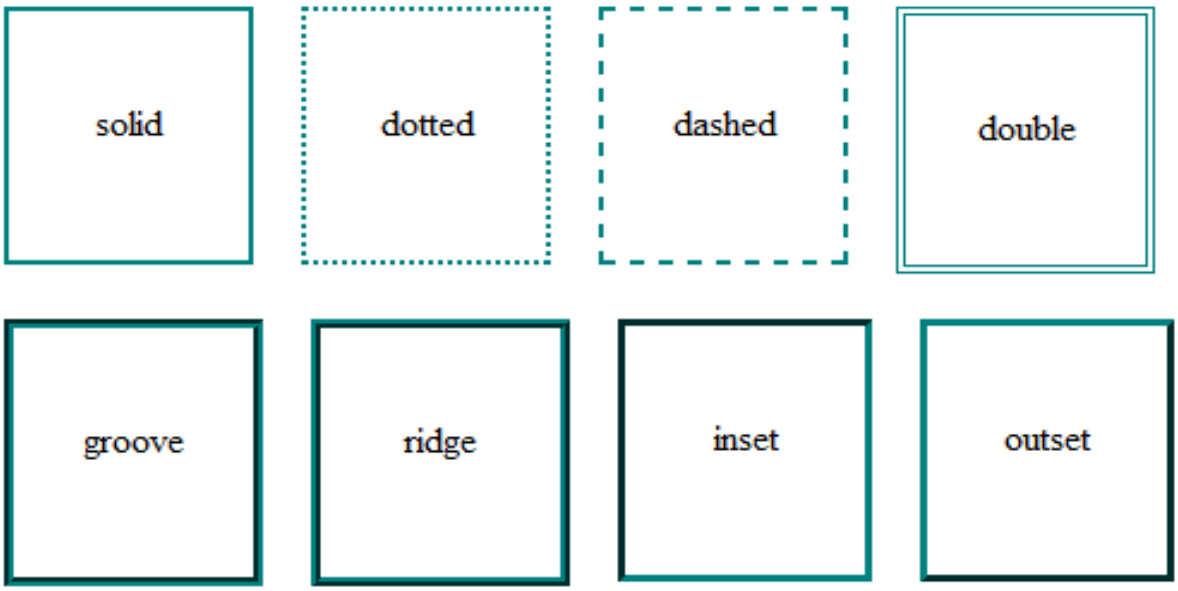
Ici encore, le code peut être simplifié

```
border : 2px solid #fff0a3;
```

Dans ces exemples il s'agit d'une bordure encadrant les éléments mais on peut définir simplement une bordure droite, gauche, haut, bas.

```
border-top : 1px dotted #000000;
/* propriétés possibles : border-top, border-bottom, border-right, border-left */
```

Valeurs possibles pour la propriété border-style :



Source de l'image : http://fr.openclassrooms.com/