

HTML

Tp 2

HTML - CSS

Florent DEVIN

26 novembre 2007

1 Utilisation des CSS

1.1 Démonstration par l'exemple

Les feuilles de styles font partie intégrantes de la norme XHTML 1.0. Normalement, dans votre code HTML, il ne devrait pas y avoir d'attribut de mise en forme. La mise en forme ne s'effectue que dans les feuilles de style. Pour mieux comprendre l'intérêt des CSS, créez une page web dont le contenu est :

```
<ul class="menu">
  <li><a href="">Menu 1</a>
    <ul>
      <li><a href="http://www.eisti.fr">Sous menu 11</a></li>
      <li><a href="">Sous menu 21</a></li>
    </ul>
  </li>
  <li><a href="">Menu 2</a>
    <ul>
      <li><a href="">Sous menu 21</a></li>
      <li><a href="">Sous menu 22</a></li>
    </ul>
  </li>
</ul>
```

Normalement vous devriez avoir quelque chose qui ressemble à la figure 1.



FIG. 1 – Menu sans css

Rajouter les balises nécessaires pour permettre l'utilisation des feuilles de style. Il faut mettre dans l'entête du fichier HTML :

```
<link rel="stylesheet" type="text/css" href="./style.css" />1
```

Une fois ceci fait, vous n'avez plus à toucher au code HTML. Nous nous concentrerons sur les feuilles de style uniquement. Éditez donc la feuille de style `style.css`. Dans ce fichier, tapez le texte suivant, essayez de taper les blocs un par un et de visualiser le résultat intermédiaire à chaque fois :

```
ul.menu, ul.menu ul {  
    width: 80px;  
    border: 1px solid #000;  
    background-color: #8aa;  
    padding-left: 0px;  
    cursor: default;  
    margin-left: 0px;
```

¹Bien sur, si vous souhaitez utiliser une feuille de style qui s'appelle `style.css`

```
}

ul.menu li {
  list-style-type: none;
  margin: 0px;
  position: relative;
  color: #fff;
}

ul.menu li ul {
  display: none;
  position: absolute;
  top: 2px;
  left: 78px;
}

ul.menu li:hover {
  background-color: #ffa;
  color: #000;
}

ul.menu li:hover > ul {
  display: block;
}

ul.menu li a {
  color: #fff;
  display: block;
  width: 100%;
  text-decoration: underline;
}

ul.menu li a:hover {
  color: #000;
}
```

2 Découpage d'une feuille HTML

La mise en forme d'une page HTML se fait via les CSS. Cependant, pour utiliser au mieux les CSS, il faut dans les pages HTML ajouter les balises `div` ou `span`. La compréhension de ces balises est facile. Ce qui est plus délicat, c'est de réussir à n'utiliser que le strict minimum.

Classiquement, on affiche du texte de gauche à droite, puis les lignes s'enchaînent les unes sur les autres. On a donc une notion de positionnement horizontal, et une autre de positionnement vertical. La balise `span` permet d'indiquer que le positionnement de l'élément se fait en ligne. C'est à dire que plusieurs balises `span` enchaînées les unes derrière les autres s'afficheront sur une même ligne. L'utilisation de la balise `div` permet de préciser que l'affichage se fait par bloc. Les différents blocs se positionneront les uns au dessus des autres.

Créez une page web. Dans le corps de celle-ci tapez le code suivant :

```
<p><span class="gris">Ceci est un </span><span  
class="blue">exemple</span></p>
```

Avec la feuille de style suivante :

```
.gris {  
  color: gray;  
}  
  
.blue {  
  color: blue;  
}
```

Visualisez le résultat, et passez le au valideur du W3C. Maintenant changez les `span` en `div`. Visualisez le résultat. De même, passez le au valideur.

3 Positionnement des blocs

3.1 Notion de positionnement

Le plus dur à maîtriser dans les feuilles de style, c'est le positionnement des blocs. Avant de rentrer dans le détail précisons quelques notions. Pour comprendre correctement le positionnement des boîtes, il est important de

connaître la notion de flottants, ainsi que celle du positionnement absolu. Nous avons déjà vu que certaines balises "s'emboîtent" en ligne, tandis que d'autres s'empilent. C'est ce que l'on appelle le *flux normal*. La notion de flottants est un peu différente de la précédente. En effet, lorsque l'on utilise la propriété flottante, la boîte est positionnée dans le flux normal. Puis elle en est extraite pour la repousser vers le côté indiqué. Lorsque l'on utilise cette propriété le contenu des autres blocs peut "couler" autour du bloc créé. Enfin, le positionnement absolu permet d'extraire complètement le bloc du calcul du flux normal. Il est alors positionné en fonction des propriétés spécifiées.

Lorsque l'on utilise la propriété `position`, on a le choix entre `static`, `relative`, `absolute`, ou `fixed`. Si vous n'utilisez pas cet attribut, alors c'est que vous utilisez le mode de placement `static`. Voici l'explication sommaire des ces différents attributs :

static La boîte est normale, placée selon le flux normal. Les propriétés `left`, `top`, `right` et `bottom` ne s'y appliquent pas.

relative L'emplacement de la boîte est calculé selon le flux normal (ce qu'on appelle la position dans le flux normal). Ensuite la boîte est déplacée relativement à cette position dans le flux normal. Quand une boîte *B* a une position relative, l'emplacement de la boîte suivante est calculé comme si *B* n'avait pas été déplacée.

absolute L'emplacement de la boîte (et éventuellement sa taille) est déterminé par les propriétés `left`, `right`, `top`, et `bottom`. Celles-ci spécifient les déplacements en fonction du bloc conteneur. Les boîtes en position absolue se situent hors du flux normal. Elles n'ont ainsi aucune influence sur la mise en forme des autres éléments de même degré de parenté.

fixed L'emplacement de la boîte est calculé comme pour `absolute`, mais la boîte est en plus fixe par rapport à une référence donnée.

3.2 Notion de boîtes

3.2.1 Présentation

Avant de rentrer plus dans le détail, nous devons nous attarder sur la notion de boîte. La figure 2 explique les différents éléments qui régissent les boîtes.

Pour chacune des propriétés, il est possible de fixer les tailles. On utilisera les propriétés suivantes en fonction de ce que l'on désire fixer : `border`,

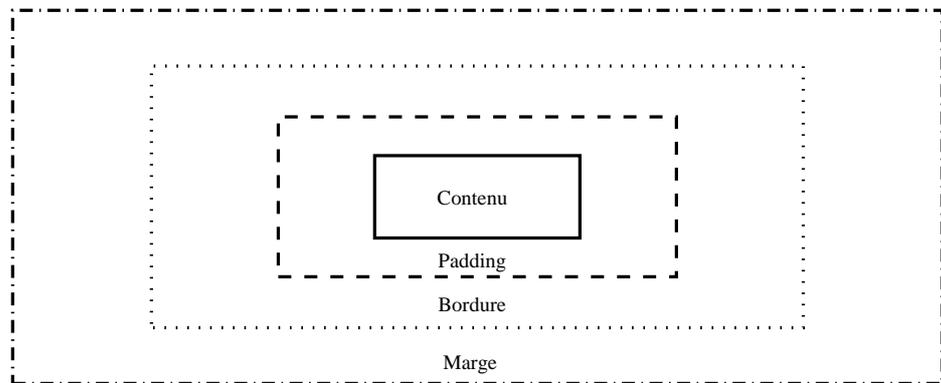


FIG. 2 – Dimensions d’une boîte

margin, **padding**. Pour chacune des ces propriétés on peut fixer les dimensions à droite, gauche, en haut, ou en bas. On utilisera alors, par exemple, **margin-top**, **margin-right**, **margin-bottom**, ou **margin-left**. Il est possible de fixer l’ensemble des valeurs en une seule fois. Il suffit d’utiliser **margin**. L’utilisation de cette propriété nécessite quelques explications. Si on écrit **margin : 2em** ; alors l’ensemble des marges sera fixé à **2em**. Si on écrit **margin : 2em 1em** ; alors les marges haut et bas vaudront **2em**, tandis que celle de gauche et de droite vaudront **1em**. Si on écrit **margin : 2em 1em 3em** ; alors la marge haute vaudra **2em**. Celle de gauche et de droite **1em**, tandis que la marge du bas sera de **3em**. Si on fixe quatre valeurs, **margin : 1em 2em 3em 4em** ; alors les marges prendront les valeurs respectives pour le haut, la droite, le bas et la gauche.

Il existe plusieurs types de valeurs. Il existe des valeurs relatives, et d’autres qui sont fixes. Parmi les valeurs relatives on trouve : **em**, **ex**, **px**. **em** signifie la largeur qu’utilise un **m** dans la police concernée. C’est souvent d’ailleurs la taille de la police. **ex** signifie la hauteur du **x** dans la police concernée. **px** signifie *pixel*. C’est théoriquement une valeur relative, car elle dépend du média de visualisation employé. On peut aussi utiliser le **%**. Dans ce cas, le **%** est relatif par rapport à la valeur du ”père”.

Parmi les valeurs fixes, on trouve : **in**, **cm**, **mm**, **pc**, **pt**. ce qui signifie respectivement *pouce* (2.54cm), *centimètre*, *millimètre*, *pica* (12 points) et *point* ($\frac{1}{72}\text{ pouce}$).

3.2.2 Attention

Il y a possibilité que les marges de boîtes attenantes fusionnent. Normalement en CSS2, les marges horizontales ne fusionnent jamais, ce qui n'est pas le cas des marges verticales. De plus, et c'est là un problème, les navigateurs ne calculent pas tous la largeur, et hauteur d'une boîte de la même façon. Certains prennent en compte les différents espacements dans le calcul de l'espace utilisé par la boîte, d'autres ne les prennent pas en compte.

Exercice 1 : Positionnement en CSS

Cet exercice à pour objectif de vous faire travailler avec les feuilles de style pour la création d'une présentation de page à 3 colonnes. Pour qu'il vous soit profitable, faites-le sérieusement, sans vous précipiter sur le résultat final qui apparaît en fin de page. Si vous voulez vous apercevoir des difficultés que l'on peut rencontrer lorsque l'on crée un site, je vous invite à faire le TP sous windows, avec IE, et Firefox.

Le fichier `exo_div.html`, contient 5 balises 'bloc' `<div>`. Chaque `div` contient un titre `<h1>` et un paragraphe de texte `<p>`.

Question 1 : Copiez le code source de ce fichier

Question 2 : Passez le code au validateur

Question 3 : Donner un identifiant (`id`) à chaque `div`

Question 4 : Passez le code au validateur

Question 5 : Créez une feuille de style et liez-la au document HTML

Question 6 : Dans la feuille de style définir, pour `body`, `padding` à 0 `margin` à 0

Question 7 : Définissez Verdana comme police de caractère et une taille très petite

Question 8 : Donner une couleur de fond différente à chaque `div` (Le texte doit rester visible)

Question 9 : Voir le résultat obtenu dans Firefox (que se passe-t-il?)

Question 10 : Définir une valeur `padding` pour chaque `div` (on peut prendre la même valeur pour tous les selecteurs)

Question 11 : Voir le résultat dans Firefox.

Question 12 : Définir une valeur `margin` pour chaque `div`

Question 13 : Voir le résultat dans Firefox.

Question 14 : Remettre tous les `padding` à 15 pixels et `margin` à 0

- Question 15** : Définir des valeurs *width* pour chaque `div` (des valeurs différentes)
- Question 16** : Voir le résultat dans Firefox
- Question 17** : Par défaut, quelle largeur les navigateurs donnent-ils à un `div` ?
- Question 18** : Par défaut, comment les navigateurs positionnent-ils les `div` ?
- Question 19** : Supprimez la *largeur* du 1^{er} `div`
- Question 20** : Atribuez une *largeur* de 200 px au 2^e `div`
- Question 21** : Atribuez une *largeur* de 200 px au 3^e `div`
- Question 22** : Supprimez la *largeur* du 4^e `div`
- Question 23** : Supprimez la *largeur* du 5^e `div`
- Question 24** : Ajoutez la règle `{float :left}` au 2^e `div`
- Question 25** : Ajoutez la règle `{float :right}` au 3^e `div`
- Question 26** : Que se passe-t-il ?
- Question 27** : Ajouter la règle `{clear :both}` au 5^e `div`
- Question 28** : Que se passe-t-il au niveau des navigateurs
- Question 29** : Quelle type de présentation obtenez-vous ?
- Question 30** : Passez votre code HTML au validateur
- Question 31** : Passez le Css au validateur. Résolvez les problèmes.