PeiP 2 / 2016 - 2017

Dino Lopez, Benjamin Miraglio et Julien Philip

Introduction aux technologies du Web

***Techniques avancées pour le* CSS.**

Chaque élément de la page web va se voir associé une boite CSS en vue de son affichage. La boite sera gérée en *inline* ou en *block*, selon sa nature par défaut ou en se fiant aux règles CSS. Le navigateur calcule alors où l'élément sera placé dans la fenêtre, en fonction de la taille de la fenêtre et des boites précédentes déjà placées. Le placement naturel (le flux) qui s'applique se fait logiquement de haut en bas et de gauche à droite.

Astuce : il existe un outil de développement web intégré à Firefox, accessible via F12.

1 Positionnement de boîtes

Donner un format convenable à une page web peut devenir quelque peu compliqué si l'on commence à modifier de manière désordonnée le flux normal d'affichage des éléments d'une page web. Pour éviter cela, il existe des façons propres de modifier le style d'une page.

Ces différentes façons ne sont pas toujours compatibles avec tous les navigateurs. Ainsi, ce qui s'affiche correctement sur Firefox peut s'afficher mal sur Chrome.

1.1 Centrer horizontalement un élément

Pour centrer du texte dans un élément HTML, il possible d'utiliser la propriété `text-align` :

```
p.texteaucentre {
  text-align: center
}
```

On utiliserait alors le code HTML suivant : `<p class="texteaucentre"> Texte centré </p>`.

Pour centrer un bloc qui ne prend qu'une certaine partie de la page (par ex. 80

```
.centre {
    margin: auto;
    width: 80%;
}
```

Certaines balises HTML ne possèdent pas la propriété `text-align`. Il est alors possible de combiner ces balises avec la balise `<p>`. Par exemple, avec la balise `<img />` : `<p class="texteaucentre"><img src="fig.png" alt="une image"/></p>`.

1.2 Centrer verticalement un élément

Pour aligner verticalement un élément, on a la propriété `vertical-align`. Pour centrer verticalement un élément, il faudrait donc appliquer la règle `"vertical-align: middle;"`. Le problème est que cette propriété est applicable uniquement aux boîtes inline et aux cellules des tableaux. Comment fait-on donc pour centrer un texte (rappelant qu'un paragraphe de texte <p> est une boite inline) ?

Solution 1 : placer un tableau invisible et y mettre le texte.

Solution 2 : plus légère et simple, combiner une boîte div avec la boîte p, mais "convertir" la boîte div en cellule de tableau, grâce à la propriété display égale à table-cell.

```
div.milieu {
    min-height: 10cm;
    display: table-cell;
    vertical-align: middle;
}


<div class="milieu"><p>texte</p></div>
```

2 Création de colonnes

Lorsque vous souhaitez travailler avec des éléments placés côte à côte (afin de créer des colonnes), la première chose à faire c'est de décider la taille de ces colonnes. Il n'est pas conseillé de fixer des tailles de colonnes en terme de pourcentages car les changements de taille de la fenêtre du navigateur peuvent conduire à des mauvaises positions de colonnes.

Par exemple, si on veut créer une page web à 2 colonnes, on peut décider que chaque colonne prend 450 pixels (ce qui est raisonnable si on suppose une largeur d'écran de 1024 pixels). De plus, la première colonne devrait avoir la propriété "float : left", mais pas pour la 2ème colonne. En revanche, sur cette dernière, on devra donner la propriété "margin-left : 460px ;", ce qui donnera un espace de 10 px entre les 2 colonnes, si on suppose une bordure de 0 px. (Si le border est de 2 px pour la colonne 1, et qu'on veut tout de même 10px entre les 2 colonnes, margin-left : 464 px ; pour positionner la colonne 2).

```
.col1 {
    border: 2px solid blue;
    float: left;
    width: 450px;
    margin-left: 0px;
    margin-right: 0px;
}

.col2 {
    border: 2px solid black;
    margin-left: 460px;
    width: 450px;
}
```

Si l'on voulait 3 colonnes, ce sont les colonnes 1 et 2 qui flotteraient à gauche, sauf la dernière, qui aurait la propriété "margin-left". Vous avez sans doute compris maintenant, on fait flotter toutes les colonnes, sauf la dernière.

```
.col1 {
    border: 2px solid blue;
    float: left;
    width: 200px;
}
```

```
.col2 {
    border: 2px solid black;
    float: left;
    margin-left: 10px;
    width: 200px;
}

.col3 {
    border: 2px solid black;
    margin-left: 428px;
}
```

Néanmoins, essayons de mieux comprendre la notion de float : une boite en mode float va "flotter", cad, elle va être placée au fur à mesure du flot que construit le navigateur, au contraire des boites dont le positionnement est dicté en "dur" par la règle CSS. Ainsi, la colonne 1 va se placer le plus à gauche possible, mais comme avant elle, a priori, il n'y a rien à sa gauche, elle va se retrouver au bord gauche de la fenêtre. La colonne 2 va elle venir se placer à la suite de la colonne 1, dans le flot normal (haut bas, gauche droite, et donc, pourrait se retrouver en dessous de la colonne 1 si par hasard la fenêtre est trop étroite). La colonne 3 va elle être placée à une position fixée consistant à laisser une marge de 428 px, à sa gauche, calculée par rapport au bord gauche de la fenêtre dans notre exemple.

Question : Pourquoi un margin-left de 428px donne une distance de 10px entre la colonne 2 et 3 ?

Une fois nos colonnes créées, on devra terminer ce flux "anormal" de boîtes. Comment ? la prochaine boîte après les colonnes (qui sera une boîte en mode block) devra avoir la propriété "clear : both ;" (ce qui signifie que cet élément ne doit pas avoir d'éléments flottants à ses côtés gauche et droit).

Dans le cas où les tailles sont toutes fixes, si on veut éviter que nos colonnes ne restent entassées toutes à gauche lorsque le navigateur de l'utilisateur a une taille très supérieure à la largeur totale de colonnes, on devrait centrer (l'ensemble de) nos colonnes. Heureusement, vous savez maintenant comment centrer un élément sur une page web :)

Exercice d'application 1 : Ecrire un document html (servez-vous par exemple de `Exo1application.html`) puis le styliser à l'aide d'un fichier css, afin qu'il dispose de 3 colonnes côte à côte et centrées dans la fenêtre, puis d'un paragraphe en dessous, prenant toute la largeur de la fenêtre. Vérifier l'allure de votre page en étirant et rétrécissant la fenêtre de votre navigateur. Cette allure est résumée par l'image `Exo1Application.bmp`, où la 3eme colonne occupe tout l'espace disponible à droite, dans la fenêtre. Ce n'est pas forcément très beau dans cet exemple, mais cela pourrait être voulu.

En particulier, dans une page à double colonne, où la première est utilisée pour placer un menu vertical, il faut donner une taille fixe à la colonne menu. Doit-on aussi donner une taille fixe à la colonne de droite ? Si on le fait, le danger est d'avoir un vide à droite, si la fenêtre du navigateur est trop large. Si au contraire, on ne précise rien, alors, comme pour l'exercice que l'on vient de faire, la colonne de droite peut occuper tout l'espace à droite. Reste un souci si l'on rétrécie trop la fenêtre : la colonne de droite peut être trop étroite, et ce n'est pas beau non plus. Pour ne pas avoir ce problème, on peut ne pas donner de taille fixe, mais un min-width sur cette colonne de droite.

Exercice d'application 2 : Modifier à nouveau le précédent CSS, afin de produire l'agencement de nos 3 colonnes (cf. `Exo2Application.bmp`). Si la fenêtre de navigation est suffisamment large, on a la 3ème colonne qui occupe tout l'espace, comme à l'exercice 1. Si la fenêtre ne l'est pas assez, la colonne 3 doit rester d'une taille minimale de 200px, et si besoin un ascenseur horizontal apparait automatiquement.

Expérience supplémentaire : il est possible de positionner en fait nos 3 colonnes en les faisant toutes floater à gauche. Ceci en fixant juste une marge-left pour la 3eme colonne correspondant à l'écart voulu entre la 2ème et la 3ème, et en fixant une largeur pour la 3ème.

Variation : tester la toute nouvelle possiblité offerte en CSS3, qui consiste à créer des colonnes facilement. Notez qu'elles auront toutes la même taille ! Voici la règle CSS3 correspondante que vous pouvez tester :

```
.lescols { /* on met par ex dans un <div class="".lescols"> tout le texte à colonner */
   width: 500px; /* au max, ce texte en colonnes prendra 500px de large */
   column-width: 150px; /* selon la dim souhaitee par colonne, il en rentrera ici 3  */
   column-count: 3; /* ca tombe bien: on voudrait 3 colonnes */
   column-gap: 10px; /* l'espace entre chaque sera de 10px */
}
```

3 Création de Menus : une application du principe des colonnes

Que ce soit un menu horizontal ou un menu vertical, tout commence par placer une simple liste avec la balise `ul`. Puisque chaque entrée du menu (qui peut consister à basculer vers un sous-menu) est un lien vers une autre page web (ou section dans la page), on aura également des balises `<a>` à l'intérieur de chaque `<li>`. C'est mieux si cette liste est à l'intérieur d'un div qui nous permette de l'isoler du reste du contenu de la page web.

```
<div class="mmenu">
    <ul>
        <li><a href="page1.html">Entrée 1 du menu</a></li>
        <li><a href="page2.html">Entrée 2 du menu</a></li>
        <li><a href="page3.html">Entrée 3 du menu</a></li>
    </ul>
</div>
```

3.1 Menu horizontal

Commençons donc par déclarer la taille de notre menu. Si chaque entrée du menu prend 150px, pour 3 entrées on aura 450px. Si chaque option a un bord d'1px, alors ça fait 6px de plus. Conclusion : notre menu aura une largeur de 456px.

```
.mmenu > ul {
    width: 456px;
}
```

Notez la présence du sélecteur `>`, qui nous facilitera la création d'un sous-menu, qui se déroule vers le bas, si jamais on en a besoin. Dit autrement, ce `>` permet de sélectionner l'`ul` qui est directement dans l'élement de la classe `.mmenu`, sans risque de confusion avec d'autres éléments `ul` qui pourraient exister à l'intérieur d'items de cette `ul`.

Notre menu sera placé par défaut à gauche de l'écran. Comment faire pour centrer notre menu ?

Dans un menu horizontal, on souhaite voir le menu de la forme

```
Entrée1    Entrée2    Entrée3
```

Cependant, le mode *block* par défaut des `<ul>` fait que ça s'affiche en forme de liste et verticalement. Nous utiliserons ici à nouveau la technique de flottants pour mettre toutes les entrées du menu sur une seule ligne, vu précédemment dans la partie "Création de colonnes". Profitons également pour créer les bords de chaque entrée du menu et donner l'illusion d'avoir des boutons dans le menu.

```
.mmenu > ul > li {
    float: left;
    border:1px solid \#8AC007;
}
```

Esthétiquement parlant, c'est mieux d'avoir le texte du menu centré à l'intérieur de chaque bouton. Comment fait-on pour centrer le texte qui ira à l'intérieur de chaque option de menu ?

Au passage, comment fait-on pour supprimer le bullet correspondant à chaque `li` de l'`ul` ?

Pour rendre "beau" chaque bouton de notre menu, on pourrait ajouter une couleur de background à chaque lien de l'entrée. Mieux encore, on pourrait créer une figure en forme de bouton qui pourrait être mise en background.

Ici, on va se contenter de mettre au moins une couleur de fond. Puisque le fond du lien `<a>` doit prendre la totalité du bouton, indépendamment de la taille du texte du lien, on devrait utiliser la propriété width pour fixer la taille du bouton. Problème : width est applicable uniquement aux balises "block" (e.g. les div), or, `<a>` est une balise "inline". Transformons donc `<a>` en mode block et fixons la taille souhaitée du bouton (150px selon ce qui a été spécifié au début de cette sous-section).

```
.mmenu > ul > li > a {
    color: white;
    background-color: red;
    display: block;
    width: 150px;
}
```

Finalement, visuellement, on peut aider le lecteur de la page à savoir sur quel bouton il a placé (sans cliquer) la souris, en changeant sa couleur de background (ou en changeant l'image de fond, selon le cas). Voici le code CSS :

```
.mmenu > ul > li a:hover {
    background-color: blue;
}
```

Jusqu'à présent, vous avez un menu fonctionnel avec les effets visuels/format de base. Il pourrait être encore amélioré en donnant des bords différents pour simuler un bouton "cliqué", etc.

Et après le menu ?

Jusqu'ici, nous avons travaillé avec les boîtes "li" en mode flottant. Nous devrons donc appliquer la propriété "clear : both" dans la prochaine boîte, comme vu précédemment sur la partie "Création de colonnes", afin de rendre à notre page Web son flux normal.

Exercice d'application 3 : Mettre en place un tel menu horizontal, centré, à 3 entrées de 150px, selon l'agencement indiqué sur l'image `Exo3Application.bmp`.

3.2 Ajouter un sous-menu

Imaginez que l'option 2 de notre menu possède plusieurs options. Il faudrait donc créer un sous-menu, qui devrait être visible uniquement lorsque la souris se trouve sur l'option "menu 2". Pour créer un sous-menu, commençons par déclarer le contenu de ce sous-menu. Par exemple :

```
<li><a href="page2.html"> Entrée 2 du menu </a>
    <ul class="smenu">
        <li>entrée 1 sous-menu</li>
        <li>entrée 2 sous-menu</li>
    </ul>
</li>
```

Pour pouvoir changer les propriété de l'`<ul>` contenant le sous-menu, nous nous aiderons de la classe CSS que nous avons nommée "smenu".

La première chose à faire, est de déclarer comme invisible (avec "display : none ;") le sous-menu, et profiter également pour lui donner les bonnes dimensions. On constate donc que même si un élément HTML est présent, la prise en comtpe de sa boite CSS dans l'agencement des boites dans la fenêtre de navigation peut être rendue optionnelle.

```
ul.smenu {
    width: 150px;
    padding-left: 0px;
    display: none;
}
```

La propriété padding-left permet que chaque entrée du sous-menu soit complètement alignée avec l'option du menu principal.

Ensuite, montrons le sous-menu qui se trouve juste sous le li lorsque la souris est dessus (notez l'utilisation du sélecteur ">" après "li :hover", permettant d'être certain d'afficher le "ul" qui est directement inclus dans le li, et pas un éventuel "ul" qui se trouverait dans un sous-sous-sous...-menu), grâce à la propriété "display : block ;". Profitons également pour enlever les bullets qui précédent chaque entrée de la liste lorsque' "ul" reprend sa forme "block". Notez aussi que la pseudo classe hover peut s'appliquer à autre chose qu'un lien.

```
.mmenu ul li:hover > ul {
    display: block;
    list-style-type: none;
}
```

Et voilà, le sous-menu est maintenant prêt. Encore une fois, il y aurait plusieurs choses à faire en plus pour obtenir un résultat visuellement très agréable, mais vous avez maintenant vu le principe du support de sous-menus dynamiques, c-à-d ne s'affichant que lorsque nécessaire.

Exercice d'application 4 : Ajouter au résultat obtenu à l'exercice 3 où nous avions un menu à 3 entrées, un sous menu qui apparait dynamiquement lorsque on positionne la souris sur la seconde entrée. Ce sous menu, qui apparaît dynamiquement change d'aspect selon qu'on est sur telle ou telle de ses entrées. Ces changements d'aspects se voient au travers de changements de couleurs, selon l'agencement montré dans le fichier `Exo4Application.bmp`.

3.3 Menu vertical

Un menu vertical suit le même principe que le menu horizontal. Ceci peut même être un peu plus simple, du fait que le mode "block", par défaut pour les "ul" et "li", n'a pas besoin d'être modifié.

Savoir fabriquer un menu vertical est bien utile si l'on veut par exemple positionner le menu sur un des côtés de la page, dans une colonne, et laisser le reste occuper l'espace libre. Ceci revient à appliquer un découpage en 2 colonnes, une colonne contenant le menu, l'autre colonne le reste de la page, et en fixant les marges de ces zones de manière appropriée. Ceci est d'autant plus intéressant si on a besoin de se déplacer verticalement dans la zone de contenu, mais qu'on veut garder à tout moment le menu visible.

Exercice d'application 5 : Repartir du résultat obtenu à l'exercice 3. mais laisser le menu à 3 entrées en mode vertical comme il l'est par défaut. Le placer à gauche d'une largeur suffisante (au moins 150px). Dans une zone d'affichage, placée de manière adéquate, insérer un contenu textuel suffisamment gros. L'entrée 2 du menu permet de se placer en bas de ce contenu (grâce à une ancre interne référencée dans l'href du lien), mais pour autant, on veut continuer à voir le menu sur la gauche. L'agencement initial est donc dans le fichier `Exo5-1Application.bmp`, puis, la situation d'arrivée une fois l'entrée 2 cliquée dans le fichier `Exo5-2Application.bmp`.

Pour obtenir le résultat désiré, donc pour continuer à voir le menu à gauche une fois descendu bas dans la page, il faut fixer sa position de manière fixe par rapport à la fenêtre du navigateur. Il n'est donc pas nécessaire d'utiliser un float : left. Mais il faut faire attention à ce que les éléments de contenu soient suffisamment décalés sur la droite, grâce à un margin-left appropriée. Le plus simple est de fixer cette marge pour le body en entier, ce qui par héritage sera appliqué à tous ses éléments.

```
body {
    margin-left: 180px;
}

.mmenu {
    left: 10px;
    width: 150px;
}
```