

Exercice 9 – Structure statique en colonne avec des blocks (validation des connaissances)

Cet exercice a pour objectif de valider les connaissances acquises dans la structuration de votre document HTML et le positionnement des blocks (exercices 7 & 8). Vous devez reproduire la maquette graphique contenue dans le fichier intitulé « rendu.jpg ». Pour vous aider, une méthode, « methode.jpg » vous propose une solution d'agencement.

ATTENTION : Pensez systématiquement à utiliser des « spacer », ou séparateurs après les blocks positionnés en « float ». Cf aide. @

Une aide est disponible à la fin des consignes

Une liste de quelques propriétés CSS pour les boîtes est disponible à cette adresse :

http://www.ecoledunet.com/cours/css_modele_position.pdf

L'intégralité du design devra être produit avec des boîtes <div>, ou des blocks sémantiques HTML5 ainsi que du CSS.

Sources : Une fois l'archive décompressée vous trouverez :

- Un aperçu du résultat à obtenir : « rendu.jpg »
- Une méthode, « methode.jpg »

Notions abordées : boîtes / tailles / positionnement / marges / couleur de fond / conteneur

CONSIGNES :

TIPS : Pensez systématiquement à utiliser des « spacer », ou séparateurs après les blocks positionnés en « float ». Cf aide. @

- ➔ Reproduire la maquette graphique « rendu.jpg » en tenant compte des tailles, des positions et des marges, de chaque block.
- ➔ Si cela n'est pas déjà fait, rajouter les balises de sections HTML5 correspondantes.
- ➔ Organiser votre code, faites des sauts de lignes pour le rendre graphiquement lisible.
- ➔ Uploadez votre exercice sur le serveur, via le FTP, dans votre dossier personnel
- ➔ Affichez la page en ligne

AIDE

1. Créer une boîte

Les boîtes sont des éléments rectangulaires qui permettent d'encadrer des contenus textes/multimédias, elles n'ont sémantiquement pas de sens. Il s'agit de balises universelles.

Il s'agit d'éléments de types « blocks », contrairement au type « inline ». Cela signifie que les éléments qui succèdent à la boîte passeront à la ligne. Par défaut, les boîtes se superposent.

Une fois les contenus placés dans la boîte nous pouvons positionner cette dernière, la centrer, l'aligner à gauche ou à droite. La balise qui permet de créer une boîte s'appelle <div>.

Ex : création d'une boîte

```
<div>
  Contenu à encadrer
</div>
```

NB : Lorsque nous créons une boîte, elle n'est pas visible graphiquement dans le navigateur. Il convient donc de définir des proportions ainsi qu'une couleur ou une bordure pour pouvoir la visualiser.

2. Proportions des boîtes

Il existe principalement 2 unités de mesures qui permettent de déterminer la taille d'un élément :

Le pixel : L'unité de mesure de l'écran. **Ex :** 500px

Le pourcentage : La taille de l'élément est définie en fonction de la taille écran de l'utilisateur. **Ex :** 50%

Les 2 propriétés CSS qui permettent de définir une boîte s'intitulent « **width** » et « **height** ». Il est important de prendre en considération que la hauteur, « **height** » est également défini par la hauteur du contenu placé à l'intérieur de la boîte. Ainsi, la plupart des boîtes qui incluent le corps de texte n'ont pas de hauteur définie, puisque c'est le contenu qui déterminera la hauteur.

Ex :

```
<div style="width:250px;height:300px;">
  Contenu à encadrer
</div>
```

NB : Par défaut la largeur d'une boîte est toujours 100%.

NB : Il n'est pas possible de définir une hauteur en pourcentage. Excepté si le type de positionnement de l'élément est absolu « absolute ». Cf. Exercice suivants.

3. Centrer une boîte

Pour centrer une boîte ou un élément de type « block » il convient de coupler et de définir 2 propriétés CSS. Plus précisément d'associer les propriétés « margin-left » et « margin-right » à la valeur « auto ».

Ex :

```
<div style="width:250px;height:300px;margin-left:auto;margin-right:auto;">
    Contenu à encadrer
</div>
```

NB : Les boîtes qui n'ont pas de proportions mesurent par défaut 100% de la largeur de l'écran. Il est donc indispensable de définir une largeur pour pouvoir observer l'effet de centrage.

Il existe une autre modalité d'écriture que l'on nomme « concaténée », qui permet de regrouper les 2 propriétés « margin-left » et « margin-right » en une seule : « margin ».

Ex :

```
<div style="width:250px;height:300px;margin:auto;">
    Contenu à encadrer
</div>
```

4. Aligner des boîtes en lignes, méthode « float » et « clear »

Il existe 2 méthodes pour « coller », ou disposer plusieurs boîtes les unes à la suite des autres... Chacune de ces méthodes permet d'arriver plus ou moins facilement au même résultat et présentent des avantages et des inconvénients. A vous de choisir...

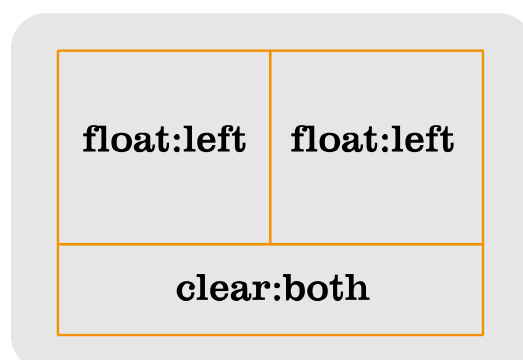
- « float » (a utiliser)

- La propriété CSS « float » doit être associée aux 2 éléments qui doivent être collés.
- Lorsqu'un élément est en « float » (flottant), il n'est plus vraiment intégré dans le flux de blocks de la page, dans la structure. Il la survole en quelque sorte.
- **L'élément qui suit une balise positionnée en float doit systématiquement avoir une propriété CSS «clear» avec la valeur « both », pour s'intégrer.**

Dans le cas contraire (si vous n'utilisez pas la propriété « clear »), l'élément remontera au même niveau que le dernier élément float et sera positionné en dessous, donc partiellement visible.

Ex : (complet d'une d'utilisation correcte)

```
<div style="width:250px;height:300px;float:left;"> collé </div>
<div style="width:250px;height:300px;float:left;"> collé </div>
<div style="width:500px;height:100px;clear:both;"> à la ligne </div>
```



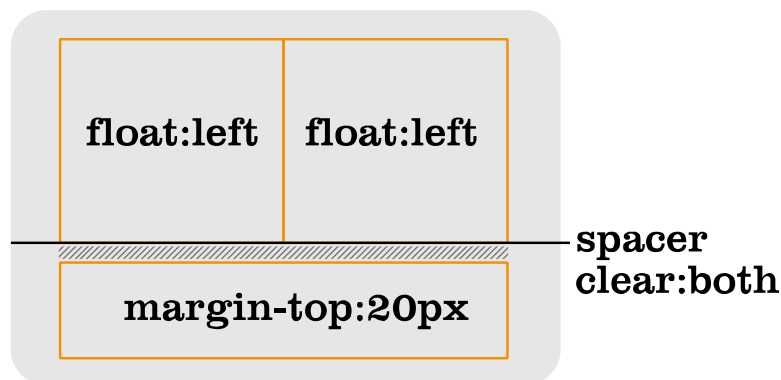
- « clear » (à utiliser)

La propriété "clear" interdit à un élément d'être adjacent à un élément positionnée en « float ». Ainsi « clear » place l'élément sous toute boîte flottante qu'elle soit à gauche ou à droite. C'est généralement la raison pour laquelle utilisée la propriété « float » implique systématiquement d'utiliser la propriété CSS « clear »

Par ailleurs, il n'est pas possible d'utiliser une marge haute ou une marge supérieure sur un élément ayant une propriété « clear ». Pour contrecarrer ce problème, nous utiliserons un « spacer ». Il s'agit d'une boîte qui ne sera pas graphiquement représentée mais qui fera office de séparation entre le l'élément en « float » et l'élément qui suit.

Ex.: (spacer)

```
<div style="width:250px;height:300px;float:left;"> collé </div>
<div style="width:250px;height:300px;float:left;"> collé </div>
<div style="clear:both;"><!-- séparateur ou spacer -->
<div style="width:500px;height:100px;margin-top:20px"> à la ligne
</div>
```



5. Centrer et gérer les marges extérieures

En CSS, la propriété qui permet de centrer un élément et de gérer les marges extérieures est la même.

Il est possible de définir indépendamment une marge extérieure pour les 4 cotés d'un élément de la manière suivante :

```
<div style="margin-top:5px;"> </div> <!-- Marge haute -->
<div style="margin-right:10px;"> </div> <!-- Marge droite -->
<div style="margin-bottom:15px;"> </div> <!-- Marge bas -->
<div style="margin-left:20px;"> </div> <!-- Marge gauche -->
```

Bien entendu il est possible de coupler ces propriétés, « margin-bottom :5px ;margin-left :10px ».

Lorsque l'on souhaite combiner plusieurs de ces propriétés de marges extérieures, on utilise la propriété « margin » qui est, souvenez-vous, également utilisée pour centrer un élément. Comment faire ?

La propriété margin permet de définir successivement la valeur des 4 marges que l'on va avoir en suivant le sens des aiguilles d'une montre.

```
<div style="margin: 5px 10px 15px 20px;"> </div>
```

Dans cet exemple, la boîte a comme propriété :

- une marge extérieure haute de 5px
- une marge extérieure à droite de 10px
- une marge extérieure en bas de 15px
- une marge extérieure à gauche de 20px

Souvenez vous également que pour centrer un élément il suffit d'attribuer la valeur « auto » à la marge gauche et à la marge droite.

```
<div style="margin: 5px auto 15px auto;"> </div>
```

Ainsi, la boîte dispose d'une marge extérieure haute de 5px, d'une marge extérieure basse de 15px et sera centrée dans la mise en page.

6. Les sections sémantiques (HTML5)

La plupart des balises HTML servent à structurer le contenu mais n'ont aucune valeur sémantique, elles sont neutres. C'est par exemple le cas des boîtes <div>. HTML5 a introduit l'utilisation de balises qui permettent de définir le rôle et le sens du contenu encadré.

Ce tableau récapitule les différentes balises qui permettent de créer des sections sémantiques ainsi que leurs spécifications.

| Nom | Détails |
|-----------|--|
| <section> | Section générique regroupant un même sujet, une même fonctionnalité, de préférence avec un en-tête |
| <article> | Section de contenu indépendante, pouvant être extraite individuellement du document ou syndiquée (flux RSS ou équivalent), |
| <nav> | Section possédant des liens de navigation principaux (au sein du document ou vers d'autres pages). Le menu |
| <aside> | Section dont le contenu est un complément par rapport à ce qui l'entoure, qui n'est pas forcément en lien direct avec le contenu. Il s'agit généralement de d'une colonne. |
| <header> | Section d'introduction d'un article, d'une autre section ou du document entier (en-tête de page). Cette partie peut avoir une partie graphique uniquement |

<footer> Section de conclusion d'une section ou d'un article, voire du document entier (pied de page).

Ex :

```
<article> Le contenu de mon article </article>
```

Sources : <http://www.alsacreations.com/article/lire/1376-html5-section-article-nav-header-footer-aside.html>

7. Optimiser la compatibilité des sections sur tous les navigateurs

Pour optimiser la compatibilité des sections HTML5 avec tous les navigateurs, vous pouvez employer cette portion de code **à coller entre les parties <head> de votre document.**

```
<script type="text/javascript">
  document.createElement("header");
  document.createElement("footer");
  document.createElement("section");
  document.createElement("aside");
  document.createElement("nav");
  document.createElement("article");
  document.createElement("figure");
</script>

<style>
header, footer, section, hgroup, aside, nav, article, figure,
figcaption, time, dialog {display: block;}
</style>
```