

Chapitre 6: Les feuilles de style – Notions avancées

Introduction

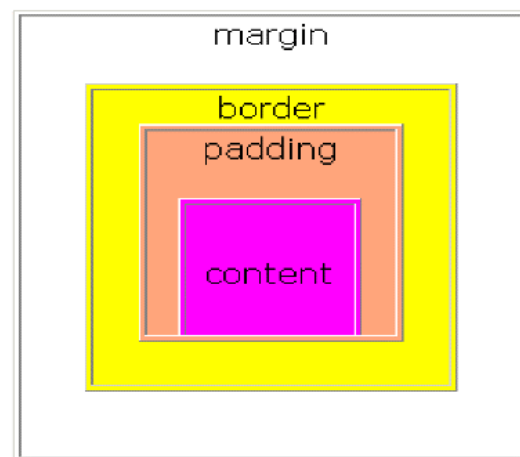
Dans ce chapitre on va aborder des notions avancées sur les feuilles de styles à savoir :

- Le modèle de boîtes
- Le positionnement
- Le flottement
- Etc.

I. Le modèle de boîtes

La recommandation **CSS1** indique que tous les éléments **HTML** (sauf ceux qui ne figurent que par leurs adresses : images, son, objets multimédia en général, plug-in, etc) puissent être considérés comme des **blocs rectangulaires**. Ce bloc est constitué de plusieurs couches ; on a de l'intérieur vers l'extérieur :

- un contenu (*content*)
- une zone d'ajustement ou marges intérieures (padding)
- un encadrement (*border*)
- une marge extérieure (*margin*)



Cette terminologie est décomposée selon le côté concerné : droite (**right**), gauche (**left**), dessus (**top**), dessous (**bottom**).

1. Règles d'attribution

Pour tous les éléments définis ci-dessus, une propriété définit son épaisseur, il s'agit :

margin	padding	border-width
margin-top	padding-top	border-top-width
margin-right	padding-right	border-right-width
margin-bottom	padding-bottom	border-bottom-width
margin-left	padding-left	border-left-width

2. Border

a. *border-width*

Les valeurs possibles de cette propriété sont une **longueur**, ou bien *thin* (mince), *medium* ou *thick* (gros). Cette propriété est compatible avec tous les navigateurs.

b. *border-color*

border-color permet de spécifier la couleur de la bordure.

La répartition des couleurs s'organise en commençant au sommet, puis en tournant dans le sens des aiguilles d'une montre. Si une couleur est manquante, on prend celle qui lui fait face. Si l'encadrement est présent mais pas la propriété *border-color*, le navigateur reprend par défaut la couleur du texte (*color*).

c. *border-style*

border-style permet de spécifier le style de la bordure.

Les valeurs possibles sont :

- *none*, la valeur par défaut (absence de bordure)
- *dotted*, bordure en pointillés
- *dashed*, bordure en tirets
- *solid*, bordure continue
- *double*, bordure double
- *groove*, bordure 3D en creux
- *ridge*, bordure 3D en saillie
- *inset*, bloc 3D en arrière
- *outset*, bloc 3D en avant
- *hidden*, bordure cachée

d. *border-top*, *border-right*, *border-bottom*, *border-left*

border-top, *border-right*, *border-bottom*, *border-left* et *border* permettent des raccourcis :

```
p.intro {
    border-top : thick blue double;
}
p.conclu {
    border-left : medium red dotted;
}
p.corps {
    border : 2px solid blue;
}
```

II. Le positionnement en CSS

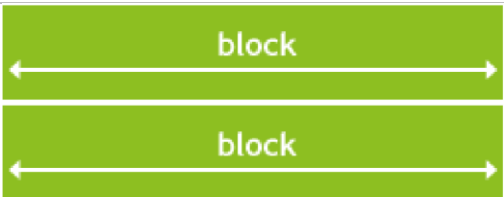


Le positionnement en CSS consiste à gérer la position des éléments les uns par rapport aux autres dans une page web.

1. Flux d'un document

Il existe en **HTML** deux types d'éléments : les éléments en ligne et les éléments de type bloc.

- ♣ **Les éléments en ligne (*inline*)** sont destinés à être placés au fil du texte. Il s'agit par exemple des éléments d'emphase (*em*), des images (*img*), des liens (*a*), etc. Normalement, ces éléments n'ont pas vocation à être placés sur une page, mais à être affichés dans le contexte de l'élément parent qui les encadre. Un élément en ligne ne peut contenir que des éléments en ligne ou du texte, et n'a pas de marge ou de padding prédéfini.
- ♣ **Les éléments de type bloc (*block*)** sont ceux qui, par défaut, ont un rendu visuel "en bloc" sur un navigateur graphique : c'est le cas par exemple des titres (*h1...6*), des paragraphes (*p*), des tableaux (*table*), etc. Ils peuvent être positionnés sur une page. Les éléments de type bloc peuvent contenir des éléments en ligne. Tous les éléments de type bloc, sauf les paragraphes et les titres, peuvent de plus contenir d'autres éléments de type bloc.

Exemple

Block		
Inline		
inline-block		

2. Positionnement

Grâce à la technique de positionnement CSS, vous aurez la possibilité de placer les éléments sur une page web en se référant à une maquette.

a. Positionnement absolu

Le positionnement *absolu* vous permet de placer un élément où vous voulez sur la page, au pixel près. Il faut en effet dire à quel endroit on veut placer l'élément sur la page, grâce aux quatre propriétés CSS suivantes :

- **top** : indique la distance par rapport au haut de la page.
- **Bottom** : indique la distance par rapport au bas de la page.
- **Left** : indique la distance par rapport au bord gauche de la page.
- **Right** : indique la distance par rapport au bord droite de la page.

Exemple

```
background-color:#ffe;
position:absolute; top:50px;
left:10px; width:200px;
```

```
background-color:#060;
color:#0f0; position:absolute;
top:20px; left:250px;
width:200px; height:125px;
```

```
background-color:#cef;
color:#639;
position:absolute; top:10px;
left:500px; width:190px;
height:180px;
```

```
background-color:#f99; color:#900;
position:absolute; top:220px;
left:10px; width:250px; height:150px;
```

```
background-color:#dfd; color:#393;
position:absolute; top:220px;
left:330px; width:250px;
height:150px; border-color:#090;
border-width:2px; border-style:solid;
padding:10px 10px 15px 80px;
```

```
background-
color:transparent;
color:#009;
position:absolute; top:450px;
left:50px; width:194px;
height:200px; border-
color:#cef; border-width:6px;
border-style:solid;
padding:12px; text-
align:center;
```

```
background-color:#eee;
position:absolute; text-align:right;
top:453px; right:70px;
width:230px; padding:10px;
```

b. Positionnement fixe

Le positionnement *fixe* est très proche du positionnement absolu. La différence réside dans le fait que l'élément reste à sa position même si on descend plus bas dans la page (d'où le nom «positionnement fixe»).

Exemple

```
.zonefixe{
  width: 50px;
  height: 50px;
  background-color: blueviolet;
  position:fixed;
  top:10px;
  left:10px;
}
```

c. Positionnement relatif

Le positionnement *relatif* est légèrement plus complexe que les autres types, il permet de placer un élément par rapport à sa position théorique dans le flux. Là encore, on doit se servir des propriétés *top*, *bottom*, *left*, *right*, mais elles ont ici une signification un peu différente.

En fait, au lieu de se rapporter aux bords de la page, elles se rapportent à la position normale de l'élément dans le flux.

Technique utile pour utiliser la superposition d'éléments avec la propriété *z-index* (voir *plus bas*).

Exemple

```
.zonerelative{
    position : relative ;
    top :50px ;
    left :80px ;
}
```

d. Positionnement static

static, la valeur par défaut. L'élément est traité "dans le flot". Il est posé dans la page à la suite des autres comme d'habitude.

Exemple

```
.zonestatique{
    position : static ;
}
```

III. Le flottement**1. float**

float permet de spécifier la position d'un texte par rapport au flux de la page, c'est-à-dire sans position « naturelle ». L'élément est alors sorti du flux. Cela permet par exemple de placer une image de manière à ce que le texte dans lequel elle se trouve la contourne. Cette propriété peut prendre les valeurs *none*, *right* ou *left*.

Exemple : Le paragraphe de classe note sera « calé » à droite de la fenêtre.

```
p.note
{
    float : right;
}
```

2. Clear (annulation du flottement)

clear permet de spécifier si l'élément peut admettre un élément « flottant » sur un de ses côtés.

Cette propriété peut prendre les valeurs **none** (la valeur par défaut), **left**, **right** ou **both**.

- Si **clear="left"**, l'élément est positionné en-dessous de tout élément « flottant » sur sa gauche.
- Si **clear="both"**, l'élément est positionné en-dessous de tout élément « flottant » sur sa gauche ou sa droite.

Exemple

```
h1
{
    clear : left;
}
```

Signifie qu'aucun élément flottant ne pourra se placer à gauche d'un titre de niveau 1.

IV. Autres propriétés

1. *overflow*

overflow permet de spécifier de quelle manière le contenu d'un élément doit être affiché s'il dépasse de la boîte qui lui est réservée. Cette propriété peut prendre les valeurs suivantes :

- *visible*, la valeur par défaut. Le contenu peut être affiché de telle sorte qu'il dépasse de la boîte réservée.
- *hidden*. Le contenu de l'élément est partiellement affiché. La position et la forme de la zone dont l'affichage est préservé sont spécifiées par la propriété clip.
- *scroll*. Des barres de défilement sont automatiquement ajoutées, que le contenu de l'élément dépasse ou non de la boîte, permettant l'affichage de la totalité de l'élément.
- *auto*. Des barres de défilement ne sont affichées que si le besoin s'en fait sentir.

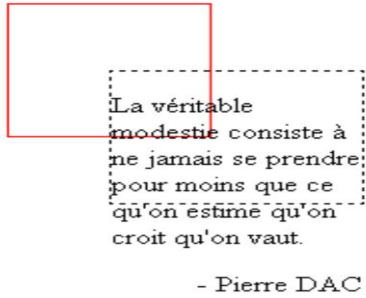
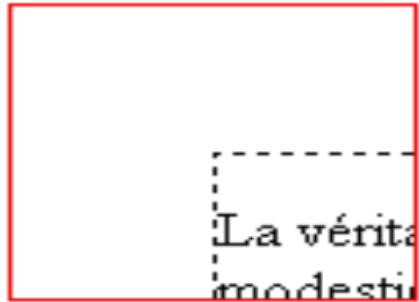

Exemple

L'élément `<blockquote>` est trop grand pour être intégralement affiché dans l'élément `<div>` parent.

```
<div>
  <blockquote>
    <p>La véritable modestie consiste à ne jamais se prendre pour moins que ce qu'on estime
    qu'on croit qu'on vaut.</p>
    <p class="auteur">- Pierre DAC</p>
  </blockquote>
</div>
```

Avec :

```
div {
  width : 100px;
  height : 100px;
  border: 1px solid red;
}
Blockquote
{
  margin-top : 50px;
  width: 100px;
  border : thin dashed black;
}
.auteur
{
  text-align : right;
}
```

Comme overflow vaut visible, la citation apparaît en entier	
Si overflow vaut hidden pour l'élément contenant, ici l'élément <div>, alors la citation est tronquée	<pre>div { width : 100px; height : 100px; border: 1px solid red; overflow: hidden; }</pre> 
Si overflow vaut scroll pour l'élément contenant, ici l'élément <div>, alors des barres de défilement apparaissent	<pre>div { width : 100px; height : 100px; border: 1px solid red; overflow: scroll; }</pre> 

2. Visibility

visibility permet de spécifier si la boîte contenant l'élément doit apparaître ou non. La place réservée à son apparition reste réservée, au contraire de la propriété **display**, qui supprime cette réservation d'espace.

Cette propriété peut prendre les valeurs suivantes :

- **visible**, la valeur par défaut. La boîte est visible.
- **hidden**, la boîte est invisible (complètement transparente), mais sa place reste réservée dans la fenêtre du navigateur.
- **collapse**, Cette valeur n'est utile que lors de la manipulation de cellules d'un tableau. Mais dans le cas général, elle est équivalente à hidden.

3. z-index

z-index permet de gérer l'ordre d'empilement des boîtes contenant les éléments.

Exemple

```
img.derriere
{
    position : absolute;
    top : 1cm;
    left : 1cm;
    z-index : 1;
}
span.devant
{
    position : absolute;
    top : 1cm;
    left : 1cm;
    z-index : 2;
}
```

Dans l'exemple précédent, la valeur du z-index de l'élément `` étant supérieure, ce dernier sera devant l'image de classe derriere. Il est possible ainsi de combiner différents niveaux de profondeurs.

4. Cursor

cursor permet de gérer l'apparence du curseur au moment où il survole l'élément. Les valeurs possibles de cette propriété sont:

- **auto**, la valeur par défaut. La forme dépend du contexte.
- **default**, le curseur par défaut du système (sous Windows, une flèche).
- **crosshair**, en forme de croix (le signe "+").
- **pointer**, la forme indiquant un lien.
- **move**, réservé normalement au cas où l'élément peut être déplacé.
- **help**, réservé normalement au cas où l'élément doit fournir une aide (en forme de point d'interrogation souvent).
- une **url**, ou une liste d'url indiquant où trouver un fichier de curseur. Si le curseur n'est pas accepté par le système, le navigateur "regarde" la deuxième adresse fournie, etc. Il est recommandé, comme pour les polices, de spécifier une forme de curseur par défaut en dernier choix.
- Etc.