Cascadind Style Sheet

Savoirs	Champs ciblé
S4.7 Langages de programmation	Web statique : HTML (3)

1. INTRODUCTION

Les feuilles de style en cascade CSS (Cascading Style Sheets) est un langage informatique qui sert à décrire la présentation des documents HTML, XHTML et XML. Les standards définissant CSS sont publiés par le W3C (World Wide Web Consortium).

2. PREFIXE DE NAVIGATEUR

Nous sommes aujourd'hui dans un entre-deux; toutes les nouveautés CSS3 ont été définies par le W3C cependant celles-ci n'ont pas encore été implémentées directement dans les navigateurs. Cependant certains navigateurs avaient développé leurs propres règles CSS3; celles-ci ont d'ailleurs servi aux recommandations du W3C.

Pour cette raison il sera nécessaire de donner une définition spécifique CSS pour chaque navigateur et que la page HTML devra toujours être testée sur le plus grand nombre de navigateur.

3. OBJECTIFS

L'un des objectifs majeurs des feuilles de style CSS est de séparer la structure d'un document de ses styles de présentation.

Avantages

- La structure du document et la présentation peuvent être gérées dans des fichiers séparés.
- La conception d'un document se fait dans un premier temps sans se soucier de la présentation, ce qui permet d'être plus efficace.
- Dans le cas d'un site web, la présentation est uniformisée : les documents (pages HTML) font référence aux mêmes feuilles de styles. Cette caractéristique permet de plus une remise en forme rapide de l'aspect visuel.
- Un même document peut donner le choix entre plusieurs feuilles de style, par exemple une pour l'impression et une pour la lecture à l'écran.
- Le code HTML est considérablement réduit en taille et en complexité, puisqu'il ne contient plus de balises ni d'attributs de présentation.

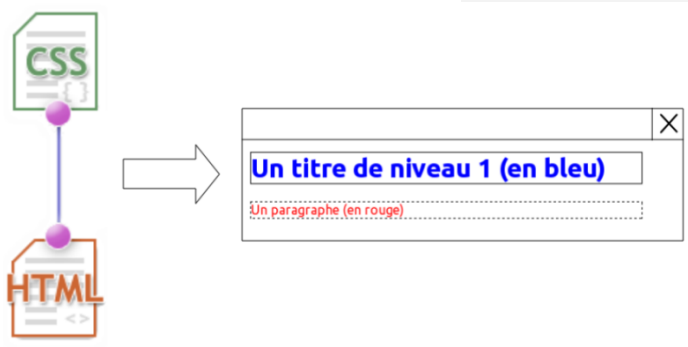
Il est par exemple possible de ne décrire que la structure d'un document en HTML :

```
<html>
   <head>
      <title>Un titre de document</title>
      <link rel="stylesheet" type="text/css" href="feuille.css" media="screen" />
   </head>
   <body>
      <h1>Un titre de niveau 1 (en bleu)</h1>
      <p>Un paragraphe (en rouge)</p>
   </body>
</html>
```

La déclaration d'une feuille CSS se fait dans l'en-tête (partie<head>) du document HTML avec la balise <link>. Il est possible de définir une feuille de style de média de sortie, voici quelques valeurs possibles de l'attribut média : all(tous), screen(écran), print(imprimante), handheld(PDA)…
Et de décrire toute la présentation dans une feuille de style CSS séparée :

```
h1 { color: blue; border: solid 1px black; }
p { color: red; border: dashed 1px black; }
```

On obtient l'affichage suivant dans un navigateur :



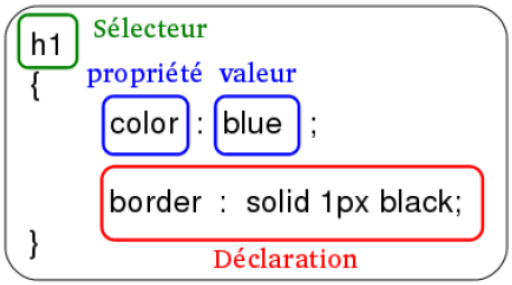
4. SYNTAXE CSS

Jeu de règles

Un jeu de règles (qualifié aussi de "**règle**") se compose d'un **sélecteur** suivi par un **bloc de déclaration** délimité par des accolades ({}).
Une déclaration est constituée d'une **propriété**, suivie du caractère deux-points (:) puis d'une **valeur**.

Les multiples déclarations qui se rapportent à un même sélecteur sont séparés par des points-virgules(;).

Une règle CSS



5. PROPRIETES ET VALEURS

Les caractéristiques de style sont exprimées sous forme de couples propriété : valeur.
Les propriétés sont libellées à l'aide de mots-outils anglais tels que « width » (largeur), « font-size » (taille de la police de caractères), ...

Les commentaires commancent par « /* » et se termine par « */ » .

6. LES SELECTEURS

Un sélecteur est un **motif de reconnaissance permettant d'appliquer un style aux éléments de l'arbre du document HTML**.

– À tous les éléments de la page : le sélecteur universel * agit sur tous les éléments HTML.

```
* { color: #ff0000; } /* pour définir une couleur rouge par défaut */
```

L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.

La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.

L'amour n'est pas la musique. La musique est la meilleure des choses. (FZ)

– À toutes les instances d'un élément : les sélecteurs de type agissent sur un type d'élément HTML.

```
/* pour définir une couleur bleue pour tous les textes des paragraphes */
p { color: #0000ff; }
```

Sauf si une autre couleur leur est attribuée de façon plus spécifique.

L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.

La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.

L'amour n'est pas la musique. La musique est la meilleure des choses. (FZ)

– À différents éléments simultanément : le regroupement des sélecteurs de type.

```
/* pour définir une couleur marron pour tous les titres de niveau 3, 4, 5 et 6 */
h3,h4,h5,h6 { color: brown; }
```

– À certaines instances d'un élément : les sélecteurs de classe.

```
/* pour définir une couleur verte pour les textes de certains paragraphes (ceux de la classe
    .vert) */
p.vert { color: #008000; }
```

Le contenu de toutes les balises <p class= « vert »> sera vert.

```
<body>
    <h1>L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.</h1>
    <h2>La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.</h2>
    <p class="vert">L'amour n'est pas la musique. La musique est la meilleure des choses. (FZ) </p>
</body>
```

L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.

La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.

L'amour n'est pas la musique. La musique est la meilleure des choses. (FZ)

– À une instance unique d'un élément : les sélecteurs d'id.

```
/* pour définir une couleur grise pour le texte d'un paragraphe précis */
p#gris { color: #7F7F7F; }
/* ou */
#gris p { color: #7F7F7F; }
```

Le contenu de la seule balise <p id= « gris »> sera gris.

– À un ensemble d'éléments successifs : aux éléments bornés par l'élément div.

```
/* pour définir une couleur verte pour une classe */
.vert { color: #008000; }

/*
qui s'appliquera aussi bien à :

<div class="vert">
   <h1>...</h1>
   <p>...</p>
   <p>...</p>
</div>

ou définis par :

<h1 class="vert">...</h1>
<p class="vert">...</p>
<p class="vert">...</p>
*/
```

– À une partie de contenu de paragraphe : l'élément span.

```
/* pour définir une couleur verte pour une classe */
span.vert { color: #008000; }

/*
qui s'appliquera à :

<p>
   <span class="vert">
      ceci sera en vert
   </span>
   ... ceci sera donc en bleu ...
</p>
*/
```

```
</head>
<body>
    <h1>L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.</h1>
    <h2>La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.</h2>
    <p>L'amour n'est pas la <span class="vert">musique.</span> La musique est la meilleure des choses. (
</body>
tml>
```

L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.

La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.

L'amour n'est pas la musique. La musique est la meilleure des choses. (FZ)

– À un élément directement ou indirectement contenu dans un autre élément : les sélecteurs descendants.

```
/* pour définir une couleur rose d'un élément em contenu dans un élément h2 */
h2 em { color: pink; }

/*
qui s'appliquera aussi bien à :

<h2>
  <em>en rose</em>
  ...
</h2>

qu'à :

<h2>
  <code><em>en rose</em></code>
  ...
</h2>
*/
```

```
</head>
<body>
    <h1>L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.</h1>
    <h2>La sagesse n'est pas la <em>vérité</em>. La vérité n'est pas la beauté. La beauté n'est pa
    <p>L'amour n'est pas la musique. La musique est la meilleure des choses. (FZ) </p>
</body>
ml>
```

L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.

La sagesse n'est pas la *vérité*. La vérité n'est pas la beauté. La beauté n'est pas l'amour.

L'amour n'est pas la musique. La musique est la meilleure des choses. (FZ)

– À un élément directement contenu dans un autre élément : les sélecteurs d'enfant.

```
/* pour définir une couleur jaune d'un élément em descendant de l'élément p */
p>em { color: yellow; }

/*
qui s'appliquera seulement à :

<p><em>en jaune</em>
   ...
</p>

mais pas à :

<p><q><em>...</em></q>
   ...
</p>
*/
```

```
</head>
<body>
    <h1>L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.</h1>
    <h2>La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.</h
    <p>L'amour n'est <em>pas la musique</em>. La musique est <q><em>la meilleure</em></q> des choses.
</body>
ml>
```

L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.

La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.

L'amour n'est *pas la musique*. La musique est "*la meilleure*" des choses. (FZ)

– À un élément suivant un autre élément : les sélecteurs d'enfants adjacents.

```
/* pour définir une couleur kaki d'un élément p suivant directement un élément img */
img + p { color: khaki; }

/*
qui s'appliquera à :

<img>...</img>
<p>en kaki</p>

mais pas à :

<h3>...</h3>
<p>...</p>
*/
```

```
<body>
    <h1>L'information n'est pas la connaissance. La connaissance n'est pas la sagesse.</h1>
    <h2>La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.</
    <img>L'amour n'est pas la musique</img>. <p>La musique est la meilleure des choses. (FZ) </p>
</body>
tml>
```

L'information n'est pas la connaissance. La connaissance n'est pas la sagesse

La sagesse n'est pas la vérité. La vérité n'est pas la beauté. La beauté n'est pas l'amour.

L'amour n'est pas la musique.

La musique est la meilleure des choses. (FZ)

7. ANCETRES, PARENTS, ENFANTS ET FRERES

Chaque document HTML est toujours composé de conteneurs. Ceux-ci peuvent être ancêtres, parents, enfants ou frères :

– Un élément **Ancêtre** est un élément qui contient un élément ou une hiérarchie d'éléments.

– Un bloc **Parent** est un élément contenant directement un autre bloc. Par exemple, un DIV contenant un paragraphe P. Attention : si ce paragraphe contient lui-même des éléments (par exemple STRONG), DIV ne sera pas Parent de l'élément STRONG mais uniquement son Ancêtre. Le Parent est donc l'Ancêtre immédiat.

– Un bloc contenu directement dans un autre bloc est dit **Enfant** de cet élément. Par exemple, ici les éléments LI sont enfants de leur conteneur UL.

– Les éléments ayant le même élément Parent sont appelés **Frères**.

Notion d'héritage

Les éléments enfants héritent de certaines valeurs de leurs éléments parents dans l'arbre du document.

Chacune des propriétés définit si elle est héritée, ou non.

Par exemple ici, tous les descendants de l'élément BODY auront la valeur de couleur 'black', la propriété 'color' étant héritée :

```
BODY { color: black; }
```

8. UNITES DE MESURES

Les CSS proposent 8 unités possibles pour exprimer tailles et longueurs (sans compter les pourcentages).
Le **W3C** les répartit en unités absolues et unités relatives :
– Les **unités absolues** sont : le point (pt), le pica (pc), le centimètre (cm), le millimètre (mm) et le pouce (in). Toutes ces unités sont équivalentes parce que proportionnelles entre elles (1 in = 2,54 cm = 25,4 mm = 72 pt = 6 pc). Le W3C précise que ces unités ne sont utiles que si les propriétés physiques du média de sortie sont connues et ce n'est jamais le cas pour les écrans.
En pratique, ces unités ne sont utilisées que pour une impression sur papier.
– Les **unités relatives** sont : le « em » (em), le « ex » (ex), le pourcentage (%) et le pixel (px). Les deux premières unités sont relatives à la police de référence : 1 em est égal à la taille de cette police, tandis que 1 ex est la hauteur du « x » dans cette police (c.à.d. celle d'une lettre sans jambage). Le rapport em/ex dépend donc de la police utilisée. Le W3C parle d'unité « relative » pour dire que le résultat physique dépend du contexte.

> Le W3C range le pixel par les unités relatives sous le prétexte qu'il n'y a pas de taille bien définie puisqu'il varie d'une machine à une autre. Certains experts préfèrent parler d'unité « fixe » pour le pixel.
> Pour deux raisons : avec des tailles de pixels, la taille du texte affiché ne tiendra pas compte des préférences de l'utilisateur, et le texte ne s'adaptera donc pas à la configuration et aux besoins des utilisateurs. D'autre part, certains navigateurs ne permettent pas le redimensionnement « à la volée » d'un texte dimensionné en pixel.

9. FLUX NORMAL

Le flux normal (ou courant) d'un document peut se définir comme étant le comportement naturel d'affichage des éléments d'une page web. Autrement dit, les éléments s'affichent dans l'ordre où ils sont déclarés dans le code HTML : verticalement, commençant en « haut » de l'écran pour aller jusqu'en « bas », et horizontalement de gauche à droite, sur la totalité de l'espace disponible et nécessaire en largeur comme en hauteur.
On distinguera deux groupes d'éléments :
– Les éléments de type **block** (div, p, h1, h2, h3, h4, h5, h6, ul, ol, li, dl, dd, table, blockquote, pre, address, etc.)
– Les éléments de type **inline** (span, a, img, span, em, strong, cite, code, abbr, etc.)
Un élément de type **bloc** (*block*) se différencie d'un élément de type **en-ligne** (*inline*) sur les principaux points suivants :
– Il occupe la totalité de la largeur de son conteneur
– Il permet l'attribution de marges verticales
– Il permet la modification de sa hauteur et largeur
Sauf exceptions, les éléments de type **en-ligne** n'occupent que la place minimum nécessaire à leur contenu.

```
<html>
   <head>
      <title>Boîtes de type bloc en flux normal</title>
      <style type="text/css">
      .rouge {background-color: #ff9999;}
      .vert {background-color: #99cc99;}
      .bleu {background-color: #9999cc;
      width: 50%; height: 50px; margin-top: 50px; /* propriétés applicables p
      }
      </style>
   </head>
   <body>
      <p class="rouge">Une boîte rouge</p>
      <p class="vert">Une boîte verte</p>
      <p class="bleu">Une boîte bleue</p>
   </body>
</html>
```

Une boîte rouge

Une boîte verte

Une boîte bleue

```
<html>
   <head>
      <title>Boîtes de type en-ligne en flux normal</title>
      <style type="text/css">
      .rouge {background-color: #ff9999;}
      .vert {background-color: #99cc99;}
      .bleu {background-color: #9999cc;
      width: 50%; height: 50px; margin-top: 50px; /* propriétés ignorées pour un élément inl
      }
      </style>
   </head>
   <body>
      <p>
         <span class="rouge">Une boîte rouge</span>
         <span class="vert">Une boîte verte</span>
         <span class="bleu">Une boîte bleue</p>
      </p>
   </body>
</html>
```

Une boîte rouge Une boîte verte Une boîte ble

On peut modifier ce comportement avec la propriété « display ».

```
<html>
   <head>
      <title>La propriété display</title>
      <style type="text/css">
      .rouge {background-color: #ff9999; display: block;}
      .vert {background-color: #99cc99; display: block;}
      .bleu {background-color: #9999cc; display: block;
      width: 50%; height: 50px; margin-top: 50px; /* propriétés maintenant applicable
      }
      </style>
   </head>
   <body>
      <p>
         <span class="rouge">Une boîte rouge</span>
         <span class="vert">Une boîte verte</span>
         <span class="bleu">Une boîte bleue</p>
      </p>
   </body>
</html>
```

Une boîte rouge

Une boîte verte

Une boîte bleue

10.PREMIER PAS AVEC LE CSS3

Mettre en forme un texte en CSS3

> Il est très important de bien choisir le formatage des polices de caractères utilisées sur une page web. En effet, pour une majorité de sites, le principal contenu est sous forme textuelle. Si ce texte est mal adapté, peu lisible, le message ne peut évidemment pas passer, quelle que soit sa qualité.

Choix des polices de caractères

Notions importantes

Avant de choisir sa, ou plutôt ses, polices de caractères pour sa page web, il faut bien connaître certaines contraintes :

- Toutes les polices de caractères n'existent pas sur tous les ordinateurs.
  - **Conséquence** : si l'ordinateur de votre visiteur ne possède pas la police que vous avez choisie, une autre s'affichera ruinant probablement le design que vous espériez.
- Mac et PC ne possèdent pas toujours les mêmes noms de polices, ni même la même résolution.
  - **Conséquence** : bien vérifier les équivalences entre Mac et PC

- Tous les navigateurs ne gèrent pas le rendu des polices de la même façon.
  - **Conséquence** : là encore, n'espérez pas avoir le même rendu partout.

Polices génériques

Il existe 5 familles de polices de caractères dites "génériques" :

- Serif
- Sans-serif
- Monospace
- Cursive
- Fantasy

Voici un test établi sur différents navigateurs.

Avec Firefox

- Serif
- Sans-serif
- Monospace
- Cursive
- Fantasy

Avec Microsoft Internet Explorer 7

- Serif
- Sans-serif
- Monospace
- Cursive
- Fantasy

Avec Opera

- Serif
- Sans-serif
- Monospace
- Cursive
- Fantasy

Avec SeaMonkey

- Serif
- Sans-serif
- Monospace
- Cursive
- FANTASY

On constatera peu de différences avec les familles serif, sans-serif et monospace, cursive n'a pas le même aspect sous PC et sous Mac.
On se contentera si possible de ce genre de polices afin de garantir une bonne homogénéité en ce qui concerne l'utilisation sur différentes machines.
Les familles de polices se déclarent à l'aide de l'attribut *font-family*, à déclarer dans le *body* pour que toute la page en bénéficie **(voir notion d'héritage)**.
On finit toujours la liste de fontes déclarées par sa famille générique à laquelle elle appartient. Ainsi, si la police déclarée n'existe pas sur l'ordinateur de votre visiteur, le navigateur affichera la police appartenant à la même famille.

Famille Serif

Les fontes de caractères de la famille serif assez courantes sont :

- Times new roman (PC)
- Times (Mac)
- Georgia (Mac/PC)
- Palatino Linotype (PC)
- Palatino (Mac)

Famille sans-serif

- Verdana (Mac/PC)
- Arial (Mac/PC)
- Trebuchet (PC)
- Helvetica (Mac)
- Tahoma (PC)
- Geneva (Mac)

Exemple :

```
body {
 font-family:"times new roman", times, serif;
}
```

Unité de taille de caractères

Il faut donc utiliser des unités relatives, telles que les *em* ou les *%*. Ces unités sont proportionnelles à la taille en pixels déclarée dans le navigateur. Par défaut, ceux-ci sont en général réglés à 16px. C'est donc une taille qui peut être modifiée par l'utilisateur... On n'a aucun pouvoir là-dessus.

La taille des caractères se déclare par l'attribut *font-size*.

```
body {
 font-family:arial, sans-serif;
 font-size:100%;
 }
h1 {
 font-size:200%;
 }
#footer p {
 font-size:90%;
 }
```

Ici, seuls les paragraphes du * * auront une taille plus petite que les autres paragraphes de la page.

Le choix des couleurs

Les couleurs se déclarent grâce à l'attribut *color*, et à l'aide de codes hexadécimaux ou rvb.

```
body {
 font-family:arial, sans-serif;
 font-size:100%;
 color:#000000; /*code hexadécimal du noir*/
 }
```

Autres attributs

Les autres attributs possibles pour modifier les caractères sont :

- Normal
- *Italique*
- PETITES CAPITALES
- **gras**

font-style: normal | italic : normale | italique
font-variant: normal | small-caps : normale | petites capitales
font-weight: normal | bold : normal ou gras

text-align: left | right | center | justify : aligné à gauche | aligné à droite | centré | texte justifié
text-decoration: none | underline | overline | line-through | blink : rien | souligné | surligné | rayé | clignotant
text-transform: none | capitalize | uppercase | lowercase : met en majuscule la 1ère lettre d'un mot | met en majuscules | met en minuscules

- surligné
- souligné
- ~~rayé~~
- clignotant (à éviter, c'est pénible... et ne fonctionne ni avec Internet Explorer, ni avec Safari)

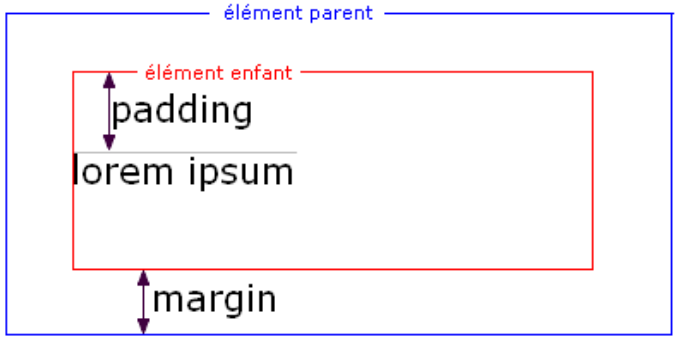
line-height: normal | nombre | %

- hauteur de ligne normale
- hauteur de ligne de 50 pixels
- hauteur de ligne de 200%

Gérer les marges

Définitions des marges intérieures et extérieures

Pour chaque élément html on peut donc définir l'espacement qui le séparera des autres éléments (*margin*) et les espacements intérieurs dont il peut bénéficier (*padding*).




```
<blockquote>
 <p>
 Lorem ipsum dolor sit amet,
 consectetur adipiscing elit. Duis eget ligula quis libero mollis dapibus.
 Suspendisse potenti. Nullam facilisis neque et sem.
 Proin placerat adipiscing urna.
 Aenean sollicitudin.
 </p>
</blockquote>
```

Blockquote est l'élément parent, *p* est l'élément enfant. (Balise *blockquote* est utilisée pour les blocs de citations)

Déclaration des tailles

Les tailles de ces marges peuvent se déclarer en pixels (*px*), en *em*, en *%*, etc. Tout dépend si l'on veut qu'elles soient fixes ou proportionnelles.
On peut détailler les tailles des marges à l'aide des suffixes *-top* (haut), *-right* (droite), *-bottom* (bas), *-left*(gauche), ou synthétiser les quatre d'un seul coup (la première valeur étant celle du haut, puis on tourne dans le sens des aiguilles d'une montre).

```
margin:2px 5px 2em 0;
```

Revient à :

```
margin-top:2px;
margin-right:5px;
margin-bottom:2em;
margin-left:0;
```

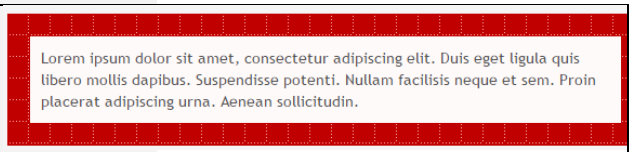
```
padding:2px 5px;
```

Revient à :

```
padding-top:2px;
padding-bottom:2px;
padding-right:5px
padding-left:5px;
```

Exemple :

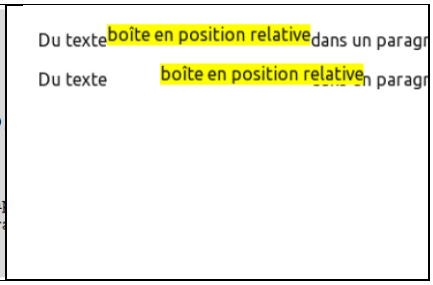
```
blockquote {
 margin:0;
 padding:1px;
 background:#C00000 url(images/quadrillage.png)
 }
p {
 margin:20px;
 padding:10px;
 background-color:#FFFAFA;
 }
```



Position des blocs

La propriété position permet de gérer les positions lorsque les possibilités du flux normal ne suffisent plus.
Le **positionnement relatif** (position : relative) permet d'inscrire un contenu en flux normal, puis de le décaler horizontalement et/ou verticalement. Le contenu précédent et suivant n'est pas affecté par ce déplacement, ce qui peut donc entraîner des chevauchements.

```
<html>
   <head>
      <title>Position relative</title>
      <style type="text/css">
      .jaune { position: relative; bottom: 5px; background-color: yellow; }
      .yellow { position: relative; bottom: 5px; left: 3em; background-color: #ffff00
      </style>
   </head>
   <body>
      <p>Du texte<span class="jaune">boîte en position relative</span>dans un paragra
      <p>Du texte<span class="yellow">boîte en position relative</span>dans un paragr
   </body>
</html>
```

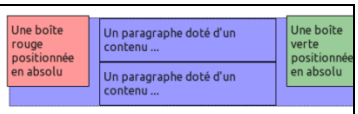


Le **positionnement absolu** (position : absolute) « retire » totalement du flux le contenu concerné :
Sa position est maintenant déterminée par référence aux limites du conteneur (c'est à dire le plus souvent la fenêtre du navigateur).

Un élément bénéficiant d'une position absolue ne bougera pas de sa position initiale tant que l'une des propriétés top, bottom, left ou right n'a pas été précisée ; il s'agit d'ailleurs là d'un comportement applicable à toutes les positions.

Un élément positionné en absolu se réfère non pas à son parent direct, mais au premier ancêtre positionné qu'il rencontre. L'élément, n'étant plus dans le flux normal, perd une de ses caractéristiques majeures qui est celle de recouvrir la totalité de la largeur disponible de l'élément parent.

```
<html><head><title>Position absolue</title>
     <style type="text/css">
     p { margin: 0; padding: 10px 5px; border: solid 1px #000;}
     .rouge { background-color: #FF9999; position: absolute; top: 1%; left: 1%; width: 20%
     .verte { background-color: #99CC99; position: absolute; top: 1%; right: 1%; width: 20
     .bleue { background-color: #9999FF; padding: 0 25%; border: dotted 1px #000;}
     </style>
  </head>
  <body>
     <div class="bleue">
        <p>Un paragraphe doté d'un contenu ... </p><p>Un paragraphe doté d'un contenu ...
        <p class="rouge">Une boîte rouge positionnée en absolu</p>
        <p class="verte">Une boîte verte positionnée en absolu</p>
     </div>
  </body>
</html>
```

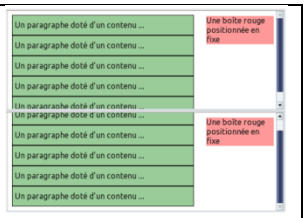


Le **positionnement fixe** (position : fixed) s'apparente au positionnement absolu, à l'exception des points suivants :

– Lorsque le positionnement est précisé (top, bottom, left ou right), l'élément est toujours positionné par rapport à la fenêtre du navigateur

– L'élément est fixé à un endroit et il ne bouge plus de la position initialement définie, même lors de la présence d'une barre de défilement.

```
<html>
  <head>
     <title>Position fixe</title>
     <style type="text/css">
     p { margin: 0; padding: 10px 5px; border: solid 1px #000;}
     .rouge { background-color: #FF9999; position: fixed; top: 5%; right: 1%; width: 25%;}
     .vert { background-color: #99CC99; margin: 0 0; width: 70%;}
     </style>
  </head>
  <body>
     <div class="rouge">
     Une boîte rouge positionnée en fixe
     </div>
     <div class="vert">
     <p>Un paragraphe doté d'un contenu ... </p><p>Un paragraphe doté d'un contenu ... </p>
     <p>Un paragraphe doté d'un contenu ... </p><p>Un paragraphe doté d'un contenu ... </p>
     <p>Un paragraphe doté d'un contenu ... </p>
     </div>
  </body>
</html>
```



Les blocs positionnés en "absolu" ou "fixé" sortent du flux naturel et ils ne sont alors plus dépendant des éléments frères. Pour se placer, un tel bloc se réfère non pas à son parent direct, mais au premier ancêtre positionné qu'il rencontre.