Feuilles de style (CSS)

v1.0

SOMMAIRE :

Qu'est-ce que les CSS ?

Pourquoi utiliser des feuilles de style ? **C**ascading **S**tyle **S**heet (CSS)
Le principe de base est le suivant : il faut séparer le contenu de la page, de son apparence. La page html contient l'information, et non la façon dont l'information est affichée. Pour un unique contenu : plusieurs affichages sont possibles. On peut penser à des affichages monochrome, sur de petits écrans, oral (le contenu de la page web est lu), une impression papier, impression sur des transparents, impression en braille…

VERSIONS DE CSS

• CSS1 publié en 1996 .
• CSS-P (CSS positioning) permet le positionnement d'éléments.
• CSS2 est une recommandation de mai 1998 qui inclut les attributs des deux précédentes versions. L'accent a été mis sur l'accessibilité et la capacité à avoir un style différent selon les media.
• CSS3 : encore en cours de construction. Nouveautés : multicolonnage, SVG, styles sur les polices (embossage...), arrondir les coins des boîtes, utiliser des images de fond dans les bordures, des ombres portées...

QU'EST CE QU'UN STYLE ?

Les styles regroupent différents attributs, tels que les choix de police, de taille, de couleur à appliquer à des titres, des sous-titres … C'est ce que va faire le CSS : permettre de définir ces attributs dans la page html à l'aide d'un code séparé. Par exemple, les balises html comme <h1>…</h1> qui par défaut possèdent une taille ou une graisse non modifiable en html, verront ces caractéristiques complètement redéfinissables en CSS.

PROPRIETES DES CSS

Police : taille, style, couleur.
Texte : alignement, casse, interlignage, espace entre les mots et les lettres.
Arrière-plans : couleur ou images.
Bords et marges : marges, remplissages, largeur, hauteur.
Bordures : style, épaisseurs, couleurs.
Interface : puces, indentation, curseur.
Positionnement : emplacement exact sur l'écran.
Affichage et visibilité : si un élément apparaît et comment.
Impression : comment définir l'aspect de la page à l'impression.

Définition des règles CSS

IL Y A 4 TYPES DE REGLES

• **Les sélecteurs html**. Il est possible de redéfinir les balises html en CSS.

• **Les classes**. Il s'agit de règles librement choisies qu'on peut appliquer à n'importe quelle balise html.

• **les pseudo-classes de lien**.

• **Les ID**. À peu près le même principe que les classes, mais ne peuvent s'appliquer qu'une seule fois dans une page.

Toutes les règles ont la même structure :

- Sélecteur (balise html, classe, pseudo-classe ou ID)
- Propriétés qui identifient ce qui est défini.
- Valeurs données à la propriété.

La paire propriété-valeur est appelée "**définition**"

selecteur { propriété : valeur ;}

```
h1 {
      font-size: 11px;
}
```

OU PLACER LES REGLES CSS?

Il y a 3 façons d'insérer les CSS :

- dans une balise html dans le corps du document.
- dans l'en-tête du document (balise <head>).
- relié au document html à l'aide d'une balise <link ..> également dans l'en-tête.

Le navigateur lit la page, télécharge la feuille de style et l'utilise pour afficher le reste de la page.

Dans la balise html

```
<h1 style="font-size: 15px; color: #555;">Titre</h1>
```

Dans l'en-tête

```
<style type="text/css">
h1 {
      font-size: 20px;
}
</style>
```

Dans un fichier externe relié au document :

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

rel="stylesheet" dit que c'est un lien externe du type feuille de style ***type="text/css"***. Le " **/>**" est utilisé si vous choisissez le XHTML 1.0.

CREATION DE LA FEUILLE EXTERNE.

À l'aide d'un logiciel de texte basique (Notepad, NotePad++ sur PC ou TextEdit, TextMate, TextWrangler sur Mac), créer un fichier enregistré au format texte seul, portant l'extension **.css**.

Aucune balise html avec les signes < > ne doit figurer dans ce fichier !

Syntaxe des règles CSS

Contrairement au HTML qui n'était pas très standardisé, les CSS ne laissent pas de place aux erreurs de syntaxe. Les navigateurs ne sont pas indulgents comme pour les fautes de html.
Chaque règle doit contenir un sélecteur et une déclaration.

Les propriétés de la déclaration doivent êtres séparées par des points-virgules.
Lorsque la valeur d'une propriété est une unité de mesure et sa valeur, il ne doit pas y avoir d'espace entre les deux :
ex:

```
font-size: 12px ;
font-family: Verdana, sans-serif;
```

Redéfinir une balise html :

```
body {
      background: #fff;
      font-family: Verdana, Times, sans-serif;
}
```

CASSE

Les CSS ne sont pas sensibles à la casse mais par convention, on met tout en minuscules.
Le W3C recommande en outre d'écrire les codes hexadécimaux en minuscules également.

CREER UNE CLASSE PERSONNALISEE

Une classe personnalisée va être utilisée ponctuellement dans les pages du site. Le nom de la classe doit être précédé d'un point, ne doit pas commencer par un chiffre et ne doit contenir ni espaces, ni accent, ni caractère de ponctuation, ni tiret.

Exemple pour une classe qui met du texte en rouge sur un paragraphe :

CSS :

```
.txtrouge {
      color: red;
}
```

HTML :

```
<p class="txtrouge"> mon paragraphe en rouge</p>
```

La valeur de la propriété de couleur peut être un mot-clef (red, green, transparent, …), un code Hexadécimal (ex: #ff56de) ou des valeurs RVB (Rouge, Vert, Bleu) entre 0 et 255 (ex: color: rgb(255,125,32);).

> NOTE : L'indication hexadécimale d'une couleur peut parfois être raccourcie en regroupant par paire les lettres et chiffres. Ainsi, "*#ff00ee*" pourra s'écrire "*#f0e*". Mais "*#ff00de*" ne pourra pas être raccourcie.

Les classes personnalisées peuvent s'appliquer à toutes les balises html.

Pour appliquer plusieurs classes à une même balise :

CSS :

```
.txtrouge {color: red;}
.italique {font-style:italic}
```

HTML :

```
<p class="txtrouge italique">mon paragraphe en rouge</p>
```

Les deux classes sont séparées par un espace, l'attribut "class" n'est indiqué qu'une seule fois.

COMMENTAIRES

```
/* Ceci est un commentaire de feuille de style */
```

Ils peuvent être multi-lignes

```
/*
Ceci est un commentaire de feuille de style
un commentaire sur plusieurs lignes
*/
```

Class et id

DIFFERENCE ENTRE CLASS ET ID

Ce sont tous les deux des sélecteurs qui permettent d'appliquer des règles à un élément.
"id" définit un élément de **manière unique**, alors que "class" peut être utilisé plusieurs fois dans la page.
Nous allons donc privilégier "id" pour les boîtes à positionner, car deux éléments ne devraient logiquement pas avoir la même position dans l'interface. De plus on peut ainsi donner des paramètres propres à chaque élément (typo, taille, couleur, positionnement, arrière-plan, bordure ...) avec un seul id.
Il est possible de cumuler id et class. Alors l'id est plus fort que la classe en cas de conflit. De plus JavaScript peut manipuler les balises avec un id.
Rappelons pour terminer qu'au sein de la CSS, les id sont définis avec le signe dièse (#nom) et les class avec un point (.nom).

Exemple de CSS :

```
/* Définition des balises HTML */
body {
	background-color: silver;
}
p {
	color: #000;
	font-family: Verdana, Arial, sans-serif;
	font-size: 12px;
}

/* Définition des ID */
#container {
	width: 950px;
	background-color: #fff;
	margin: 0 auto;
}

/* Définition des classes */
.txtImportant {
	color: red;
	font-style: italic;
}
```

Cascade, conflits et héritages

Les ID l'emportent sur les "class". Les "class" l'emportent sur les sélecteurs contextuels, pseudo-class et pseudo éléments. La feuille interne l'emporte sur la feuille externe si elle est placée après dans l'ordre du code. C'est toujours le dernier appelé qui l'emporte.

CONFLITS

```
p { color : red; }
p { color : green; }
```

c'est le vert qui l'emporte. Il écrase la valeur "red" précédemment définie.

HERITAGE

C'est le mécanisme par lequel les styles s'appliquent à un élément, mais aussi à ses descendants.

Structure d'une page html : en haut de la hiérarchie : le html qui contient body qui contient d'autres éléments.

HTML est l'ascendant de tout (aussi appelé élément racine). BODY est l'ascendant de tout ce que montre le navigateur.

Souvent les éléments héritent des propriétés de leur parent : la couleur d'arrière-plan du body sera la même pour toutes les couleurs d'arrière-plan de la page, par défaut. Quelques propriétés ne sont pas héritées, en général il s'agit de bon sens (comme border par exemple).

Unités de longueur

UNITES DE LONGUEUR ABSOLUES - (A EVITER)

- **cm** (centimètre = 1cm = 10 mm) Risque d'affichage différent sur plusieurs écrans.
- **mm** (millimètre) Même remarque.
- **pc** (pica = 1pc = 12pt) Un autre terme de typographie, donc même remarque.
- **pt** (point = 1pt = 1/72 in) Le point est une mesure typographique utilisée par les imprimeurs (d'où les "points par pouce" de certaines résolutions) et les traitements de texte.

> NOTE : Il ne peut y avoir d'espace à l'intérieur de cette valeur et pour les nombres décimaux, la virgule est impérativement remplacée par un point.

En accessibilité, ces valeurs ne sont pas recommandées, car elles empêchent l'internaute de redimensionner la taille d'affichage des caractères.
Ces valeurs sont donc à éviter côté web ! (Mais restent éventuellement utilisables pour une CSS dédiée à l'impression).

UNITES DE LONGUEUR RELATIVES - (RECOMMANDEES)

- **px** (pixel)
- **em** (largeur de la letttre m, 1 em correspond à 100% de la taille en cours de la police, 1.2 em à 120 %, 0.8 em à 80%… Son usage est donc limité aux polices.)
- **ex**(hauteur de la lettre x. Assez pauvrement implémentée, donc à éviter.).
- **%** L'élément défini prendra un pourcentage donné de la taille de son élément parent.

- **small**
- **x-small**
- **xx-small**
- **large**
- **x-large**
- **xx-large**
- **medium**
- **larger**
- **smalle**r

> NOTE : Là encore, il ne peut y avoir d'espace à l'intérieur de cette valeur et pour les nombres décimaux, la virgule est impérativement remplacée par un point.

En accessibilité, ces valeurs sont recommandées, car elles permettent à l'internaute de redimensionner la taille d'affichage des caractères.

Il est conseillé de se cantonner aux unités **%** et **em** pour un affichage fluide ou pour du texte, et à **px** quand on a besoin de placer les éléments au pixel près… Les autres valeurs seront le plus souvent évitées.

> NOTE : Il existe 2 écoles sur l'utilisation de ces valeurs. Certains intégrateurs préfèreront le "em" et en mettront partout, d'autres préfèrent les "px" et n'utilisent que ça.

Propriétés

PROPRIETES DU TEXTE

font-family:	*type de la police*	Nom(s) de(s) la police(s) (faire une liste, noms séparés par une virgule)
font-weight:	*graisse*	normal, **bold**, **bolder**, lighter, ou 100,200,300,400,500,**600,700,800,900**
color:	*couleurs*	valeurs hexadécimales ou nommées : red
font-size:	*tailles*	xx-small = xx-small x-small = très petit small = petit. medium = moyen. large = grand. smaller = plus petit que normal. larger = plus grand que normal. x-large = très grand xx-large = géant Donner des valeurs en point (pt), pica (pc), pouce (in), millimètre (mm), centimètre (cm), pixel (px), em : en relation avec la largeur de la lettre **M** de la police, ex : en relation avec la largeur de la lettre **x** % : pour pourcentage par rapport à la norme de l'élément
font-style:	*style*	normal, *italic*, *oblique*
font-variant:	*normal ou petite cap*	normal | small-caps
line-height:	*interlignage*	en points (pt), pixels (px), em ou en %
text-align:	*alignement*	left | right | center | justify
text-transform:	*Normal, nom propre, majuscules, minuscules*	None | Capitalize | UPPERCASE | lowercase
word-spacing:	*espacement des mots*	normal | ici 0,5cm entre les mots
letter-spacing :	*espacement des lettres*	normal | i c i : 0 , 5 c m
text-indent:	*indentation de la 1e ligne*	longueur (en cm ou en px) | pourcentage positif ou négatif (signe -) Effet d'un alinéa.

PROPRIETES DES LISTES

Par défaut les listes simples (non numérotées) sont précédées d'une "puce" : •

Il est possible de les remplacer avec la propriété list-style-type:

disc (par défaut)
- disc
- disc

circle
- circle
- circle

square
- square
- square

lower-alpha
(des lettres en minuscules)
a. 1er element
b. 2e element

upper-alpha
(des lettres en majuscules)
A. 1er element
B. 2e element

georgian
ა. georgian
ბ. georgian
გ. georgian
დ. georgian

armenien
Ա. armenian
Բ. armenian
Գ. armenian

cjk-ideographic
一. cjk-ideographic
二. cjk-ideographic
三. cjk-ideographic

hebrew
א. hebrew
ב. hebrew
ג. hebrew

hiragana
あ. hiragana
い. hiragana
う. hiragana

hiragana
い. hiragana-iroha
ろ. hiragana-iroha
は. hiragana-iroha

katakana
ア. katakana
イ. katakana
ウ. katakana

katakana-iroha
イ. katakana-iroha
ロ. katakana-iroha
ハ. katakana-iroha

lower-greek
α. lower-greek
β. lower-greek
γ. lower-greek

lower-latin
a. lower-latin
b. lower-latin
c. lower-latin

lower-roman
i. lower-roman
ii. lower-roman
iii. lower-roman

none
(il n'y a plus rien devant les éléments de la liste)
none
none
none

upper-latin
A. upper-latin
B. upper-latin
C. upper-latin

upper-roman
I. upper-roman
II. upper-roman
III. upper-roman

Les raccourcis

Dans le but d'alléger vos pages CSS, il est intéressant d'utiliser des raccourcis (shorthand en anglais)
Les valeurs "raccourcies" sont séparées par des espaces !

POUR LE TEXTE

font-size-adjust: none;
font-stretch: normal;
font-weight: normal;
font-style: normal;
font-variant: normal;
font-size: 16px;
line-height: normal;
font-family: sans-serif
1em/1.2em donne la taille puis l'interlignage

font: 16px Verdana, sans-serif;
ou
font: bold 1em/1.2em Verdana, sans-serif;

POUR LES LARGEURS DE MARGES ET REMPLISSAGE

Pour une même valeur en haut, à droite, en bas et à gauche, dans cet ordre précis (Top, Right, Bottom, Left - moyen mnémotechnique pour ceux qui parlent anglais TRBL, trouble ;-))

- padding : 2px
- margin : 2px

Deux valeurs différentes seront pour haut et bas, gauche et droite
margin-top : 2px;
margin-right : 5px;
margin-bottom : 2px;
margin-left : 5px;

margin: 2px 5px;

Trois valeurs différentes seront pour haut , gauche et droite puis bas
margin-top : 5px;
margin-right : 2px;
margin-bottom : 7px;
margin-left : 2px;

margin: 5px 2px 7px;

Enfin il est possible d'indiquer à la suite les 4 valeurs pour haut, droite, bas et gauche
margin-top : 5px;
margin-right : 2px;
margin-bottom : 3px;
margin-left : 6px;

margin: 5px 2px 3px 6px;

LES BORDURES

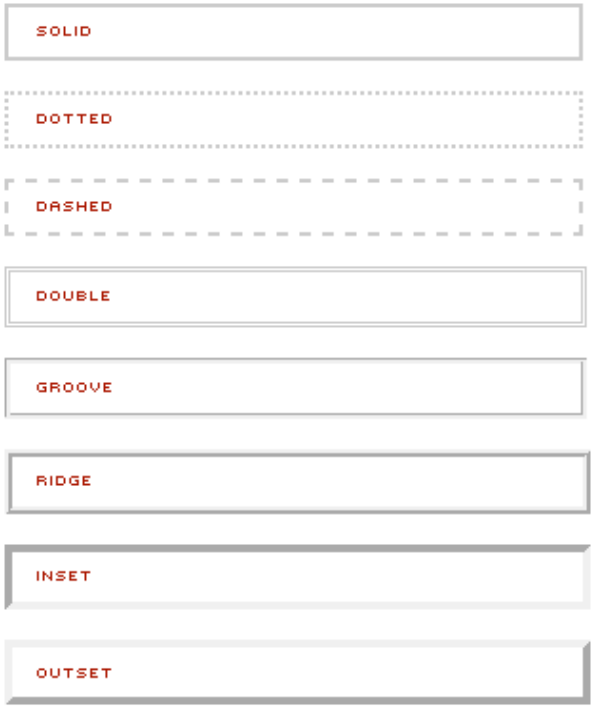
Les bordures doivent obligatoirement avoir 3 paramètres : style, largeur et couleur. L'ordre n'importe pas.

border-width: 2px;
border-style : solid;
border-color : #f00;

border: 2px solid #f00;

Les styles disponibles sont les suivants :

- solid (trait plein)
- dotted (pointillés)
- dashed (tirets)
- double (double trait plein)
- groove (bordure en relief s'enfonçant)
- ridge (bordure en relief sortant)
- inset (effet de relief s'enfonçant)
- outset (effet de relief sortant)



On peut très bien définir une bordure différente par côté de la boite !

LES COULEURS

Les couleurs peuvent être raccourcies à condition que les chiffres/lettres soient doublés. On peut donc avoir 3 ou 6 caractères dans le code hexadécimal.

color: #ffffff; **color: #fff;**
color: #00ffcc; **color: #0fc;**
color: #ee11ff; **color: #e1f;**

Les boîtes : remplissage et marges

LES MARGES

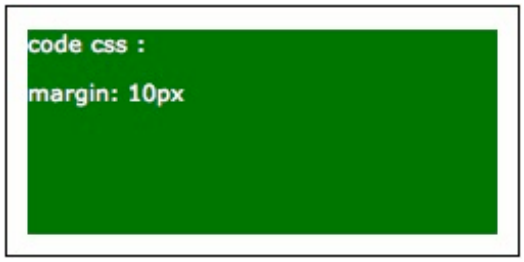
Les marges sont à l'extérieur d'un élément (boîte ou autre)

Il s'agit de l'espace qui se trouve autour de la boîte ou d'un autre élément (paragraphe, h1, listes ...)

L'arrière-plan des marges est toujours transparent. Les règles d'hérédité ne sont pas applicables.

Valeurs :

- La longueur pour définir une largeur fixe.
- Le pourcentage qui est calculé par rapport à la largeur du bloc conteneur de la boîte générée.
- Auto qui correspond à une valeur définie par le navigateur.:



La boîte verte a une marge de 10 pixels tout le tour, de son conteneur (bords noirs).

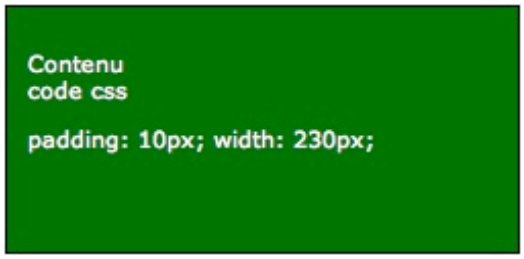
Le contenu colle aux bords

La boîte verte a une marge de :
10 pixels en haut
20 pixels à droite
5 pixels en bas
15 pixels à gauche

LE REMPLISSAGE (PADDING)

Le remplissage (padding) est à l'intérieur de la boîte.

Valeurs possibles : pixels ou pourcentage.



Ici le contenu commence à 10 pixels, la boîte prend toute la largeur.

Attention : le remplissage fait partie de la boîte et augmente donc sa largeur.

Ici la boîte prend 250 pixels de large, mais dans le code on a : 230px + 10px de remplissage gauche + 10px de remplissage droit ! (10px tout autour)

Les marges peuvent également prendre une valeur négative.

Les boites

DEFINITION

Une boîte est un morceau de la page, une "division" qui est appelée avec la balise . Chaque boîte étant un conteneur, on peut y mettre tous les autres éléments HTML, listes, en-têtes, images, flash, paragraphes, etc…

Dans le CSS, cette balise est appelée avec un "#" si c'est un id, et si c'est une classe, avec un ".".

Exemple de code html

```
<div id="page">....</div>

<div class="titre">...</div>
```

Exemple de code CSS correspondant

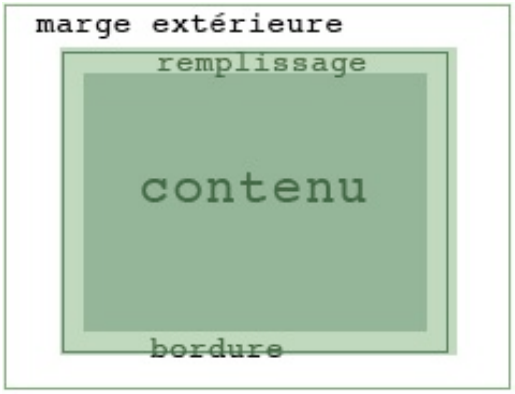
```
#page {
        width: 200px;
        font: 11px Verdana, sans-serif;
}
.titre {
        font-size: 20px;
}
```

Il y a deux modèles de boîtes : le modèle standard w3c et le modèle Microsoft Internet Explorer. Les affichages peuvent donc différer de beaucoup. Cependant si vous utilisez un doctype, cela forcera en partie IE à se comporter selon les normes du W3C.

Une boîte est composée de

- Son contenu
- Son remplissage (marge interne)
- Sa bordure

La marge externe est à l'extérieur de la boîte, mais pousse donc les éléments qui seraient adjacents.



LE POSITIONNEMENT DES BOITES : STATIQUE, RELATIF, ABSOLU

NOTION DE FLUX

Les navigateurs "lisent" le code et l'interprètent de façon linéaire. Les boîtes viennent naturellement les unes en dessous des autres, comme les paragraphes, les balises d'en-têtes et les listes, et d'autres éléments au contraire, comme les images ou les liens se placent naturellement dans le flux du document.

STATIQUE (PAR DEFAUT)

Ici il y a une boîte verte insérée dans la page, avec les marges de la page par défaut. Puis une deuxième boîte rouge, qui se place par défaut en dessous. Elles contiennent une ligne de texte.

Par défaut les boîtes font 100% de leur conteneur et prennent la hauteur de leur contenu.

RELATIF (DANS LE FLUX) = POSITION : RELATIVE

Il est possible de décaler horizontalement et/ou verticalement les éléments. Ceci peut générer des chevauchements !

Ici la boîte rouge a été "poussée" de 20pixels plus bas que l'élément précédent :

```
#rouge {
	background: red;
	position:relative;
	top:20px;
}
```

ABSOLU (HORS DU FLUX) = POSITION : ABSOLUTE

Il est possible de retirer les boîtes du flux normal de la page. Celles-ci n'auront alors aucun effet sur les autres éléments de la page. Elles se placent par rapport à l'élément parent. Si rien n'est précisé, ce sera par rapport à la fenêtre. Les boîtes positionnées en absolu prennent la largeur et la hauteur de leur contenu, sauf si on leur donne des paramètres différents.

Attention cela peut provoquer des chevauchements, comme dans l'exemple ci-dessous. La boîte rouge "ignore" la boîte verte et se positionne à 20 pixels du haut de la fenêtre.

Ici on a simplement changé la position "relative" de la boîte rouge, en "absolute" :

```
#rouge {
	background: red;
	position: absolute;
	top: 20px;
}
```

LE POSITIONNEMENT DES BOITES : LE FLOTTEMENT

Une boîte flottante est retirée du flux normal, et placée le plus à droite (float: right) ou le plus à gauche (float: left) possible de son conteneur (par défaut la fenêtre).

COMPRENDRE LE FLOTTEMENT

L'élément flottant se met le plus à droite ou à gauche possible de son conteneur, donc l'élément suivant se positionne à côté.

Les éléments flottants doivent OBLIGATOIREMENT avoir une LARGEUR DEFINIE.

Une image et du texte à côté par défaut donnent ceci :

En botanique, le lotus, dont le nom dérive du grec lotos par le latin lotus, désigne diverses plantes, arbres, arbustes ou herbes, terrestres ou aquatiques, qui portaient déjà ce nom dans l'Antiquité.

Le texte se positionne à côté de l'image, en bas et va en dessous à la fin du conteneur

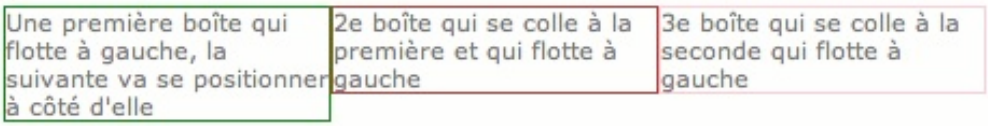
En botanique, le lotus, dont le nom dérive du grec lotos par le latin lotus, désigne diverses plantes, arbres, arbustes ou herbes, terrestres ou aquatiques, qui portaient déjà ce nom dans l'Antiquité.

Ici, l'image a le paramètre : float: left
L'image se place le plus à gauche possible et le texte l'habille donc à droite.

En botanique, le lotus, dont le nom dérive du grec lotos par le latin lotus, désigne diverses plantes, arbres, arbustes ou herbes, terrestres ou aquatiques, qui portaient déjà ce nom dans l'Antiquité.

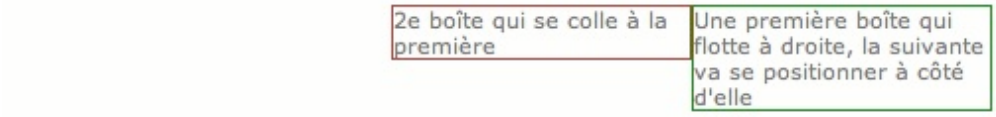
Ici, l'image a le paramètre : float: right
L'image se place le plus à droite possible et le texte l'habille donc à gauche.

FAIRE DES COLONNES



Attention les éléments flottants se placent les uns à côté des autres tant qu'ils ont la place, ensuite ils vont en dessous.

Ci-dessous les deux boîtes sont float: right, **elles partent donc de la droite du conteneur.**

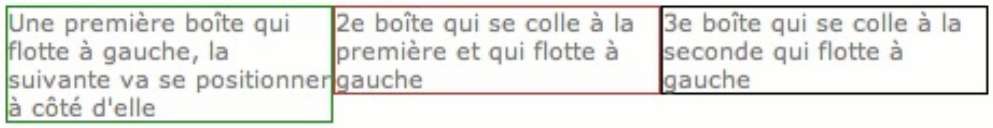


Il est recommandé de mettre toutes les boîtes qui doivent être les unes à côté des autres en float, même la dernière ! tous les navigateurs ne réagiraient pas de la même façon.

SORTIR DU FLOTTEMENT = CLEAR : LEFT, RIGHT OU BOTH

Puisque tous les éléments qui se positionnent les uns à côté des autres doivent avoir la propriété "float" y compris le dernier, il faut ensuite "forcer" le retour à la ligne.

• *Revenir à la ligne après des colonnes : clear (left, right, both) :*



Il est nécessaire d'avoir un élément qui permet de revenir à la ligne.

S'il y a eu du flottement à gauche, il faut : clear: left

S'il y a eu du flottement à droite, il faut : clear:right

S'il y a eu du flottement à gauche et à droite, il faut : clear:both

- Exemple de code html :

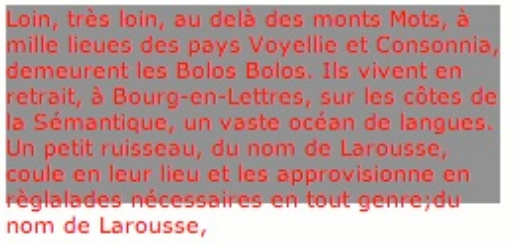
```
<div id="leftcol">colonne de gauche</div>
<div id="midcol">colonne du milieu</div>
<div id="rightcol">colonne de droite</div>
<div id="footer">pied de page</div>
```

- La CSS sera :

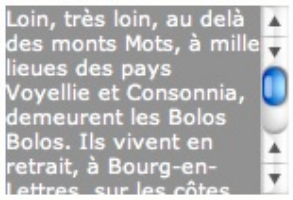
```
#leftcol    { width: 100px; float: left; }
#midcol     { width: 400px; float: left; }
#rightcol   { width: 100px; float: left; }
#footer     { width: 600px; clear: left; }
```

LA NOTION DE DEBORDEMENT : OVERFLOW

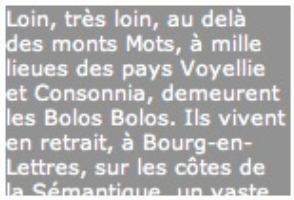
Il s'agit de la propriété "overflow" qui prend 4 valeurs : auto, visible, hidden, scroll. Par défaut le contenu débordera de la boîte et sera visible (overflow: visible).



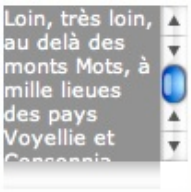
La boîte grise fait 100px de haut, mais le contenu a forcé sa hauteur, le texte dépasse de la boîte.



La boîte grise fait 100px de haut, mais nous avons ajouté la propriété : overflow: auto, cela a généré une barre de défilement de façon automatique dans le sens où il y en a besoin (verticalement et/ou horizontalement selon le contenu).



La boîte grise fait 100px de haut. Nous avons ajouté la propriété : overflow: hidden, cela a masqué le contenu qui débordait.



La boîte grise fait 100px de haut nous avons ajouté la propriété : overflow: scroll
Barres de défilement dans les deux sens (actives ou non) !

> **Attention !**
> Selon les navigateurs (Mozilla), des éléments flottant à l'intérieur d'un autre élément ne forcent pas la hauteur de celui-ci.
> Si vous donnez une couleur d'arrière-plan à une boîte qui contient des éléments flottants, il est recommandé de donner à celle-ci la propriété : overflow :auto, pour la forcer à suivre son contenu.

- Exemples de code :

html :

```
<div id="conteneur">
      <div id="col_g">colonne de gauche</div>
      <div id="col_d">colonne de droite</div>
</div>
```

CSS :

```
#conteneur  { width: 700px; overflow : auto; background: #099; }
#col_g      { width: 300px; float: left;}
#col_d      { width: 400px; float: left;}
```

Arrière-plan : images

Les images d'arrière-plan sont des images (gif ou jpg) qui par défaut se répètent en mosaïque (effet papier peint).

En CSS il est possible de ne pas les répéter, de les répéter horizontalement ou verticalement, et si on ne les répète pas, on peut les positionner dans leur élément.

Ces images peuvent s'appliquer aux balises html, que ce soit <body> pour toute la page, ou , <p>, <hn> etc ...

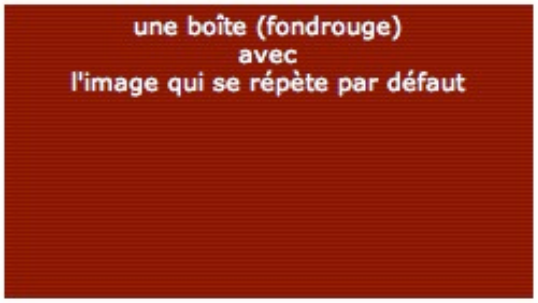
Il est recommandé d'utiliser des images les plus légères possibles et profiter ainsi de l'effet de répétition pour des effets graphiques.

Voici quelques exemples avec les codes :

- Répétition,
- Répétition horizontale,
- Répétition verticale,
- Pas de répétition,
- Remplacement de la puce de liste.

REPETITION

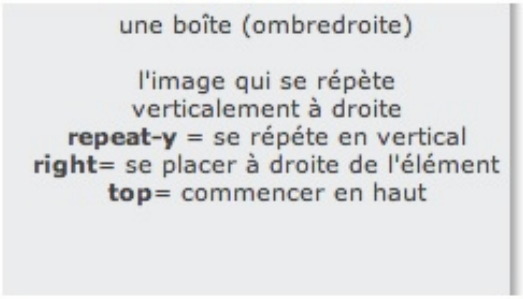
Avec cette petite image : 4px sur 2px, qui se répète par défaut




```
#fondrouge {
	background: url(fondrouge.gif);
	height: 100px;
	width: 240px;
	color: #fff;
	text-align: center;
}
```

REPETITION HORIZONTALE

Avec cette petite image : (poids: 1ko) qui se répète verticalement (cette image aurait pu ne faire qu'un pixel de haut, mais pour les besoins de lisibilité sur cette page, elle est plus grande !) De plus ici on a ajouté une couleur d'arrière-plan = #E6E7E8 :




```
#ombre-droite {
	background: #e6e7e8
	url(ombre_portee_droite.gif) repeat-y right top;
	height: 100px;
	width: 240px;
	color: #333;
	text-align: center;
}
```

REPETITION VERTICALE

Avec cette petite image :  11px sur 13px, qui se répète horizontalement :

une boîte (symboleh)

avec
l'image qui se répète horizontalement

```
#symboleh {
	background: url(symbole_h.gif) repeat-x ;
	height: 130px;
	width: 240px;
	color: #333;
	text-align: center;
	padding-top:10px;
	border:1px solid green;
}
```

PAS DE REPETITION

Avec cette image : 

une boîte (lys)

avec
l'image qui ne se répète pas
et est placée en bad à droite

Comme il s'agit d'un arrière-plan, le texte bine sûr passe par dessus

Attention à la lisibilité !

```
#lys {
	background: url(lys.gif) no-repeat right bottom;
	height: 230px;
	width: 240px;
	color: #333;
	text-align: center;
	float: left;
	padding-top:10px;
	border:1px solid gray;
}
```

REMPLACEMENT DE LA PUCE DE LISTE

Avec cette image :  on peut remplacer facilement des puces de liste par une image, et la placer exactement où l'on veut par rapport aux éléments de la liste :

une boîte (puces) avec une liste

- pommes
- poires
- oranges
- figues
- bananas

```
#puces li {
	background: url(../img/puce.gif) no-repeat left center;
	padding-left: 20px;
	list-style: none;
}
```

Les possibilités de positionnement : background-position

background-position : top, center, left, bottom, right, ou avec des valeurs en pixels ou en pourcentage.

Image d'arrière-plan : background-attachment

scroll : l'image défile en même temps que la page

fixed : l'image reste fixe et la page défile par dessus.

Pseudo Classes de liens

Par défaut les liens sont bleus et soulignés, les liens visités sont violets et soulignés et les liens actifs, rouges et soulignés.
Si vous avez paramétré une typo et une taille de texte dans votre CSS, ils vont en hériter, mais garderont les paramètres qui leur sont propres : couleur et soulignement.
Il est impératif de respecter l'ordre dans lequel ils sont appelés dans le CSS et de ne pas mettre d'espace de chaque côté des deux points.

a:link

pour le lien tel qu'il apparaît sur la page avant d'avoir été cliqué.

a:visited

pour le lien tel qu'il apparaît sur la page après avoir été visité.

a:hover

pour le lien tel qu'il apparaît sur la page lorsqu'on passe le curseur au-dessus.

a:active

pour le lien tel qu'il apparaît sur la page au moment où on clique dessus.

Pour changer la couleur : voir le chapitre sur les couleurs.

"DECORATION" DU TEXTE

Propriété text-decoration : pour la "décoration". les possibiiltés sont :

- underline (souligné = par défaut)
- overline (un trait au-dessus)
- line-through (barré)
- none (pas de soulignement)
- blink (clignotant, mais ne fonctionne pas dans Internet explorer)

Le code serait donc pour un lien non visité de couleur noire et non souligné:

```
a:link {
      color: #000;
      text-decoration: none;
}
```

Eléments de type : block

Certains éléments se mettent les uns en dessous des autres, d'autres les uns à côté des autres.
Les éléments qui se placent les uns en dessous des autres sont des éléments dits de type : **block**

les boîtes :
les paragraphes : <p>
les balises d'en-têtes : <h1> à <h6>
les listes : et
les éléments de liste :
les formulaires : <form>
les tableaux : <table>

Il sera toutefois possible de faire afficher **certains** de ces éléments les uns à côté des autres grâce à la propriété : **display**.

Le plus couramment utilisé sera de faire afficher des éléments de liste les uns à côté des autres.
Une liste ci-dessous :

- cinéma
- théâtre
- danse
- opéra

Peut s'afficher comme ceci :

cinéma théâtre danse opéra

Nous disons à la balise <li> de cette liste de s'afficher en ligne. La liste a été identifiée, par exemple **<ul id="sorties">.**
Le code CSS est :

```
#sorties li { display: inline; }
```

Pour espacer les éléments de la liste nous rajoutons :

```
#sorties li { display: inline; padding-right: 20px; }
```

Vous avez remarqué que les puces précédants les éléments de cette liste, ont automatiquement disparu !

Eléments de type : inline

Il s'agit des éléments qui se mettent les uns à côté des autres.

Les éléments qui se placent les uns à côté des autres sont des éléments dits de type : **inline** (en ligne).

les hyperliens : <a>
les images : <img>
les portions de texte:
le gras :
l'italique :
les éléments de formulaires (<input>, <textarea>, …)

Il sera toutefois possible de faire afficher ces éléments les uns en dessous des autres grâce à la propriété : **display**.

le monde le figaro libération

Les liens ci-dessus peuvent s'afficher comme ceci :

le monde
le figaro
libération

Nous disons à la balise <a> de ce paragraphe de s'afficher en *block*. Le paragraphe a été identifié, par exemple **<p id="journaux">.**

Le code CSS est :

```
#journaux a { display: block; }
```

Eléments non affichés

Il est possible de ne pas afficher des éléments.

Ceci sert :

- Lorsqu'on fait des affichages différents par type de média
- Pour des listes déroulantes de liens

On utilise encore une fois la propriété : **display**, avec cette fois le paramètre : **none**

```
#menu .sousMenu {
      display: none;
}
```

Affichage de type tableau

Il est possible d'afficher des éléments sous forme de tableau : display: table.

Les éléments imbriqués doivent avoir les propriétés **table-row** et **table-cell**, pour rappeler les bons vieux TR et TD.

La propriété **border-collapse** est utilisée pour fusionner les bordures. Elle peut prendre la valeur de :
- collapse, chaque cellule a une bordure commune.
- separate, chaque cellule a sa propre bordure.
- inherit, hérite de son parent (CSS 2)

Le style "border-collapse" ne peut s'appliquer que sur la balise HTML TABLE ou sur un élément dont la valeur display est égale : 'table' ou 'inline-table'.

Le rendu des bordures de tableau est divisé en deux catégorie dans CSS2 : fusionnées et séparées.
Dans le modèle de rendu des bordures fusionnées, les cellules adjacentes partagent les mêmes bordures.

Dans le modèle de rendu des bordures séparées, chaque cellule adjacente a ses propres bordures (la distance entre les bordures est définie par la propriété border-spacing).

Autres types d'affichage

Les autres types d'affichage décrits ci-dessous ne sont pas interprétés par tous les navigateurs, ou pas de la même façon.

DISPLAY: INLINE-BLOCK

Un élément affiché comme inline-block, est un élément en ligne, c'est-à-dire qu'il se place à côté de l'élément adjacent, mais se comporte comme un *block*.

> **Ne fonctionne pas dans Firefox 2 et inférieur.**

DISPLAY: LIST-ITEM

Signifie qu'il se comporte comme une liste, et est précédé d'une puce.

Styles divers

CURSEUR

La propriété de feuille de style *cursor* CSS peut prendre les valeurs suivantes :

Le code CSS = **cursor: nw-resize**, par exemple.

- default, curseur en forme de grosse flèche (curseur par défaut),
- pointer, curseur en forme de main avec un doigt déplié (par défaut sur les liens),
- n-resize, curseur en forme de petite flèche verticale vers le haut (North),
- s-resize, curseur en forme de petite flèche verticale vers le bas (South),
- e-resize, curseur en forme de petite flèche horizontale vers la droite (Est),
- w-resize, curseur en forme de petite flèche horizontale vers la gauche (West),
- ne-resize, curseur en forme de petite flèche en diagonale vers haut droit,
- nw-resize, curseur en forme de petite flèche en diagonale vers haut gauche,
- se-resize, curseur en forme de petite flèche en diagonale vers bas droit,
- sw-resize, curseur en forme de petite flèche en diagonale vers bas gauche,
- crosshair, curseur en forme de croix fine (genre viseur),
- move, curseur en forme de quatre flèches en croix,
- text, curseur en forme de sorte de grand I,
- help, curseur en forme de flèche et "?",
- wait, curseur en forme de sablier,
- progress, curseur en forme de flèche avec sablier,
- not-allowed, curseur en forme de rond barré (interdit),
- col-resize, curseur fait d'une barre verticale avec une flèche horizontale de chaque coté.
- row-resize, curseur fait d'une barre horizontale avec une flèche au dessus et au dessous.

```
a.aide {
      cursor: help;
}
```

Une feuille par média

Il est possible avec une seule page html, d'obtenir des affichages différents selon les types d'appareils.
Voici la liste des types existant à l'heure actuelle :
http://www.yoyodesign.org/doc/w3c/css2/media.html

all : convient pour tous les appareils (valeur par défaut si rien n'est indiqué) ;
screen : destiné principalement aux moniteurs couleurs ;
print : destiné à un support paginé opaque et aux documents vus sur écran en mode aperçu avant impression ;
aural : destiné aux synthétiseurs de parole. Voir les détails fournis dans le chapitre sur les feuilles de style auditives ;
braille : destiné aux appareils braille à retour tactile ;
embossed : destiné aux appareils à impression braille ;
handheld : destiné aux appareils portatifs (typiquement ceux avec petits écrans, monochromes et à bande passante limitée) ;
projection : destiné aux présentations en projection, par exemple avec des projecteurs ou des impressions pour des transparents ;
tty : destiné aux médias utilisant une grille de caractères fixe, tels les télétypes, les terminaux ou les appareils portatifs aux capacités d'affichage réduites. Les auteurs ne devraient pas utiliser de valeurs exprimées en pixel avec ce type de média ;
tv : destiné aux appareils du type télévision (avec ces caractéristiques : basse résolution, couleur, défilement des pages limité, sonorisé).

> NOTE : Les noms des types de médias ne sont pas sensibles à la casse.

Une première feuille de style externe est reliée à la page comme par exemple :

```
<link href="styles/ecran.css" rel="stylesheet" type="text/css" media="screen"/>
```

puis une autre :

```
<link href="styles/impression.css" rel="stylesheet" type="text/css" media="print"/>
```

L'instruction "media" définit pour quel type d'appareil chaque feuille est faite.

On peut ainsi adapter au mieux les affichages : des pixels, des polices sans-serif pour l'écran seront transformées par exemple en cm et polices serif pour l'impression. De même pour la largeur des blocs, l'affichage des menus, ou des publicités.

Ressources

LIENS :

W3C : http://www.w3.org

Pour les standards du web : http://openweb.eu.org/

La puissance démontrée des CSS avec le superbe site ZenGarden :

http://www.csszengarden.com/tr/francais/

Alsacréations (tutoriaux, forums, …) : http://www.alsacreations.com

Articles, ressources traduites en français : http://www.pompage.net/

Des galeries de sites en CSS : http://www.mostinspired.com/?type=gallery

LIVRES :

"*Mémento CSS*", de Raphaël Goetter (Eyrolles)

"*CSS 2 Pratique du design Web*", de Raphaël Goetter, Philippe Vayssière, et Elie Sloïm (Eyrolles)

"*Le Zen des CSS*", de Dave Shea & Molly Holzschlag (Eyrolles) → Bonne ressource pour contourner les faiblesses d'Internet Explorer.