



## Références :

- <http://www.w3schools.com/css/>
- <https://www.w3.org/Style/CSS/Overview.fr.html>

HTML a ses limites. La mise en page est difficile. La taille des polices est limitée. La maintenance est difficile. Le contenu d'un document HTML n'est pas séparé des balises de mise en forme, toute modification est difficile.

CSS (Cascading Style Sheet ou Feuilles de style en cascade) est une norme et une recommandation de W3C. La première version CSS1 a été créée en 1996 et la deuxième CSS2 en 1998, CSS2.1 en 2004 (avec environ 70 nouvelles propriétés). CSS3 en cours de finalisation.

Le principe d'une feuille de style CSS est de séparer la mise en forme du contenu de document. Les instructions CSS de mise en forme sont rassemblées au début de document HTML dans la balise <head> ... </head> ou carrément dans un fichier à part. Dans une feuille de style CSS nous décrivons toute la mise en forme des balises HTML.

## Où mettre une feuille de style ?

Il y a trois possibilités :

**1) CSS globale** : à inclure dans <head> ... </head> à l'aide de la balise

```
<style type="text/css"> ... règles css ... </style>
```

**2) CSS importée** : Une feuille de style est décrite dans un fichier (d'extension .css) à part et attaché au document (X)HTML à l'aide de

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

à placer dans <head> ... </head> ,

**3) CSS Intra-lignes** : réalisée à l'aide de l'attribut style, elle modifie la mise en forme d'une balise localement :

```
<balise style=" règles CSS "> ... </balise>.
```

Déconseillée.

CSS est un langage où chaque mot de son vocabulaire est appelé **propriété** et chaque propriété à un ensemble de **valeurs** possibles prédéfinies. La syntaxe est

**propriété : valeur ;**

qu'on appelle déclaration. Une propriété est séparée de sa valeur par : et chaque déclaration se termine par ; . Par exemple : color est une propriété qui prend comme valeur une couleur. On écrit alors color:red ; et pour rendre tous les titres de niveau 1 bleus, dans la feuille de style on écrit h1{color:red; }.

**Sélecteur** peut être une balise HTML, plusieurs balises HTML aux quelles on veut donner la même mise en forme, deux balises HTML avec un lien de parenté, un style qu'on définit et qu'on voudra appliquer à plusieurs différentes balises, ou autres .(voir ci-dessous).

## Structure d'une règle CSS

```
sélecteur {propriété1:valeur1 ;
           propriété2:valeur2 ;
           ... }
```

Une feuille de style est formée d'un ou de plusieurs sélecteurs. Par exemple : h1, h2 {color : blue} Pour la lisibilité, il vaut mieux écrire chaque ligne sur une ligne :

## Exemple :

```
h1 { font-family: Arial;
     font-size: 12 ;
     color: red; }
```

## Généralité sur le langage CSS

**Casse** : Le langage CSS est indépendant de la casse. En général on écrit en minuscule.

**Commentaire** : une ou plusieurs lignes /\* ... \*/ comme en C, C++, Javascript ou // pour mettre en commentaire la dernière partie d'une ligne

**Couleurs en CSS** : peuvent être déclarées de 3 manières :

- par **nom de couleur** en anglais : aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white, yellow; orange etc . . .

- **#rrggbb**, où r,r',g,g',b et b' sont des nombres hexadécimaux, rr',gg' et bb' indiquent la portion de rouge, vert et bleu respectivement.

- **rgb(r,g,b)**, où r, g et b sont soit des pourcentages soit des entiers de 0 à 255, pour le rouge, le vert et le bleu.

## Exemple :

```
h1{color:#0000ff; background-color:red;}
h1{color: rgb(50%,50%,50%);
background-color:rgb(25,125,255);}
```

## Unités de longueur en CSS

### Les unités absolues :

in : inch (pouce, 1in=2,54cm)  
pt : point (point, 1pt=1/72in, environ 0,0352 mm)  
pc : pica (1pc=12pt)  
mm : millimètre  
cm : centimètre

### Les unités relatives :

em = unité relative à la hauteur d'une police donnée  
ex = hauteur du caractère "x" minuscule d'une police  
px = pixel, fonction de la résolution de l'écran de l'utilisateur

% = pourcentage par rapport à la norme de l'élément HTML

Le séparateur **décimal** est le point et non pas la virgule.

Toute longueur en CSS **exige** une unité.

### div et span

Les balises, <div> et <span> n'ont aucune présentation particulière, et servent à s'attacher un style CSS.

**div** permet de définir un **bloc** de texte particulier.

**span** sert à définir un style pour une sous-partie d'une balise existante. s'utilise souvent avec des feuilles de style intra-lignes.

```
<p style="color: blue">Ce texte est en bleu avec une
partie en <span style="color: red"> rouge </span></p>.
```

### Sélecteur .class

Une classe en CSS sert à définir un style dans le but de l'appliquer à différentes balises.

```
.ma_classe {propriété1:valeur1; propriété2:valeur2; ...}
```

Pour appliquer ce style à une balise :

```
<balise class="ma_classe"> .... </balise>
```

```
.classe1 { background : yellow; font-color: #000080 }
```

```
<p class="classe1"> ... </p>
```

```
... <td class=" classe1"> texte </TD></TD> ...
```

Un style peut aussi être défini spécifiquement à une balise :

```
balise.nom_de_classe { propriété : valeur; ... }
```

Pour utiliser la classe nom\_de\_classe :

```
<balise class="nom_de_classe"> .... </balise>
```

Exemple :

```
.style1{color : red;}
```

```
h1.style2{color : blue;}
```

```
<h1 class="style1"> ... </h1>
```

```
<p class="style1"> ... </p>
```

```
<h1 class="style2"> ... </h1>
```

### Identifiants : #mon\_id

Pour utiliser les feuilles de style avec du Javascript on utilise les id (identifiants). Les id fonctionnent exactement comme les classes. La syntaxe est :

```
#mon_id {propriété1:valeur1; propriété2:valeur2; ... }
```

Et pour l'appliquer :

```
<balise id="mon_id"> ... </balise>
```

**Important** : on ne peut faire qu'un seul appel à #mon\_id par page.

### Sélecteur class :

On peut nommer et définir un style indépendamment de toute balise HTML et qu'on pourra appliquer à différentes balises ou donner plusieurs mise en forme à une même balise.

```
.nom_de_classe { propriété : valeur; ... }
```

### Propriétés CSS

**L'exposant** <sup>3</sup> qui suit une propriété indique que celle-ci fait partie de CSS3 alors que les autres sont de CSS1 ou CSS2.

Les propriétés non supportées par la plus part des navigateurs ne sont pas citées dans ce document.

### Propriétés de texte

**text-align** : définit l'alignement du texte, valeurs : left, right, center ou justify. Exemple : h1 {text-align: right ;}.

**text-indent** : définit un retrait dans la première ligne d'un bloc de texte spécifié en longueur ou en pourcentage (de la largeur de l'élément). Exemple : p {text-indent: 1.5cm ;}

**text-transform** : change la casse (majuscule, minuscule) du texte , valeurs : uppercase, lowercase, capitalize (met le premier caractère en majuscule) . Exemple : h1 {text-transform: capitalize ;}

**color** : change la couleur du texte.

Exemple : h1 {color: #000080 ;}

**word-spacing** : espace entre les mots. Valeurs : longueur ou normal. Exemple : p{word-spacing: 7pt ;}

**letter-spacing** : espace entre les lettres. Valeurs : unité de longueur ou normal.

Exemple : p {letter-spacing: 3pt ;}

**line-height** : interligne : espace entre les lignes du texte. Valeurs : normal, unité de longueur, ou un nombre qui sera multiplié par la taille de la police pour donner l'interligne. Exemple : p{line-height: 3 ;}

**text-decoration** : définit une décoration du texte. Valeurs possibles : none, (aucune décoration), underline (souligné), line-through (barré), overline (surligner) . Exemple : h1 {text-decoration: underline;}

**text-decoration-color**<sup>3</sup> : couleur de la décoration. Exemple : p {text-decoration: underline;

text-decoration-color: blue;}

**text-decoration-style**<sup>3</sup> : style de la décoration. Valeurs : solid|double|dotted|dashed|wavy

**text-shadow**<sup>3</sup> : texte avec son ombre. Valeurs : none | décalage\_horizontale décalage\_verticale [blur-rayon] couleur de l'ombre.

Exemple : h1 {text-shadow: 2px 4px 8px #FF0000;}

**white-space** : gestion des espaces blancs. Valeurs : normal, pre (espace blancs préservés) ou nowrap (plusieurs espace blancs sont réduits à un seul) pre-line, pre-wrap. Exemple : p{white-space: pre ;}

**text-align-last**<sup>3</sup> : contrôle l'alignement de la dernière ligne d'un texte lorsque text-align a pour valeur justify. Valeurs : auto | left | right | center | justify | start | end

### Propriétés de polices de caractères

CSS fait complètement disparaître l'usage de <font>... </font>

**font-family** : pour indiquer une ou plusieurs polices, séparées par des virgules, police précise (Arial, Times, Helvetica, "verdana", 'courier...') ou famille génériques (serif, sans-serif, cursive, fantasy, monospace), il est conseillé de toujours mettre une famille générique en dernier recours. les noms comportant des espaces doivent être mis entre guillemets. Exemple : h3 {font-family: Georgia, "Times New Roman" ;}

**font-size** : changer la taille de police. Valeurs : xx-small, x-small, small, médium, large, x-large, xx-large, ou une valeur relative à la taille actuelle larger, smaller ou taille précise en pt, in, cm, px ou pourcentage de la taille de police de l'élément parent. Exemple : p {font-size: 18pt}

**font-style** : définit le style de police. Valeurs : normal, italic ou oblique.

**font-variant** : la variante de polices valeurs possibles : normal (police normale) ou small-caps (petites capitales).

**font-weight** : change l'épaisseur de police. Valeurs pour une valeur descriptive: normal, bold , bolder,

lighter, ou une valeur numérique : (100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900). Très fin (100) jusqu'à très gras (900).

**font-stretch<sup>3</sup>** : (pour fixer l'étirement de la police. Mais ne fonctionne avec aucun navigateur ! ) Les valeurs possibles :

- narrower : assez étroit
- semi-condensed : un peu condensé
- condensed : condensé
- extra-condensed : très condensé
- ultra-condensed : extrêmement condensé
- normal : étirement habituel
- wider : étirement plus large
- semi-expanded : un peu étendu
- expanded : étendu
- extra-expanded : très étendu
- ultra-expanded : extrêmement étendu

Cette propriété n'est supportée par aucun navigateur

**font** : raccourci pour les différentes propriétés de police facultatives suivantes : font-family , font-style, font-variant , font-size, font-weight. Dans l'ordre : font-style font-variant font-weight font-size/line-height, font-family. Exemple : p{ font:italic small-caps bold 30px Georgia, serif; }

### Propriétés d'arrière-plans

**background-color** : définit la couleur de l'arrière-plan . Exemple : H1 {background-color: #00FF00} .

**background-image** : insère une ou plusieurs images d'arrière-plan . Exemple : body {background-image: url('fond1.gif'), url('fond1.gif');}

**background-repeat** : définit la façon de répéter l'image d'arrière-plan repeat (Répétition sur tout l'écran, valeur par défaut) ou no-repeat ou repeat-x (Répétition selon l'axe des X) ou repeat-y (Répétition selon l'axe des Y) body {background-image: url('fond.jpg'); background-repeat: repeat-y}

**background-attachment** : spécifie si l'image d'arrière-plan reste fixe avec les déplacements de l'écran scroll ou fixed body {background-image: url('fond1.gif'); background-attachment: fixed|scroll}.

**background-position** : spécifie la position de l'image d'arrière-plan par rapport au coin supérieur gauche de la fenêtre {top ou center ou bottom , left ou center ou right} valeurs : longueur ou %. Exemple : body {background-image: url('img.gif'); background-position: right top}

**background** : raccourci pour les différentes propriétés d'arrière-plan p{background: image.gif fixed repeat} Exemple :body{background : #00ff00 url('smiley.gif') no-repeat fixed center;}

**background-blend-mode<sup>3</sup>** :

**background-clip** : précise la région à occuper par la couleur d'arrière plan. Valeurs possibles : border-box : contenu, padding et bordure padding-box : contenu et padding content-box ; uniquement le contenu (css3)

**background-origin<sup>3</sup>** : place l'origine (coin supérieur gauche) de l'image d'arrière plan. Valeurs possibles : padding-box : place l'origine de l'image à l'origine du padding border-box : place l'origine de l'image à l'origine de la bordure.

**content-box** : place l'origine de l'image à l'origine du contenu.

**background-size<sup>3</sup>** : taille de(s) image(s) d'arrière plan. Valeurs possibles :

auto : garde la taille initiale de l'image

*longueur*: peut prendre 2 valeurs (avec unité de longueur) séparées par un espace qui indiquent respectivement la largeur et la hauteur. En cas d'une seule valeur indiquée la 2<sup>e</sup> prend la valeur auto. Il en est de même si on donne deux ou un pourcentage (de l'élément parent)

cover : l'image d'arrière plan couvre l'élément.

contain : l'image d'arrière plan est couverte par l'élément.

```
{ padding:25px;
border:10px dotted #000000;
background-image:url('img.gif');
background-origin:content-box;}
```

### Propriétés de tableau

**caption-side** : Position du titre de tableau. Valeurs :

top : en haut du tableau.

bottom : sous le tableau.

left : sous le tableau aligné à gauche.

right : sous le tableau aligné à droite.

**Exemple** :caption{caption-side:bottom;}

**table-layout** largeurs fixe/variable. Valeurs :

fixed : Les mentions de largeur priment sur le contenu des cellules.

auto : Le contenu des cellules prime sur les mentions de largeur (par défaut).

**border-collapse** : modèle de bordure de cellules mitoyennes. Valeurs :

separate : les bordures de cellules de tableau ne coïncident pas (comme en HTML).

collapse : les bordures de cellules de tableau coïncident.

**border-spacing** : espace entre les bordures de cellules adjacentes. Valeur : longueur.

**empty-cells** : Affichage ou non-affichage de cellules vides. Valeurs :

show : les bordures de cellules vides sont affichées.

collapse : les bordures de cellules vides ne sont pas affichées (par défaut).

**Exemple** : table{border-collapse:separate; border-spacing:10px 50px;}

### Propriétés de liste

**list-style-type** : type de puce ou de numérotation. Valeurs pour listes ol : decimal (1, 2, etc ...), lower-roman (i, ii, etc ...), upper-roman (I, II, etc ...), lower-alpha (a, b, etc ...), upper-alpha (A, B, etc ...), lower-latin, upper-latin, armenian, decimal-leading-zero (01, 02 etc ...), georgian (an, ban, gan, etc ...), lower-greek (alpha, beta, etc ...)

Valeurs pour listes ul : disc, circle, square ou none

**list-style-position**: retrait des éléments de liste. Valeurs : inside = puces et numérotation dans le corps de la liste ;

outside = retrait à gauche des puces et numérotation (par défaut).

**list-style-image**:url("image") image de type .gif ou .jpg comme puce de liste.

**list-style** : regroupe toutes les propriétés de liste

**Exemple** : `ul{list-style:square inside url('/images/blueball.gif');}`

### Propriétés du Curseur (cursor)

Définit un curseur pour un élément (X)HTML. le curseur prend la forme mentionnée lorsque la souris survole cet élément., Exemple : `h1 {cursor : crosshair;}`  
default : curseur standard indépendant de la plateforme.

crosshair : curseur de la forme d'une simple croix.

pointer : curseur de la forme d'une flèche.

move : curseur de la forme d'une croix signalant la possibilité de déplacer l'élément.

n-resize : curseur de la forme d'une flèche pointant vers le haut (n = nord).

ne-resize : curseur de la forme d'une flèche pointant vers le haut à droite (ne = nord-est).

e-resize : curseur de la forme d'une flèche pointant vers la droite (e = est).

se-resize : curseur de la forme d'une flèche pointant vers le bas à droite (se = sud-est).

s-resize : curseur de la forme d'une flèche pointant vers le bas (s = sud).

sw-resize : curseur de la forme d'une flèche pointant vers le bas à gauche (sw = sud-ouest).

w-resize : curseur de la forme d'une flèche pointant vers la gauche (w = ouest).

nw-resize : curseur de la forme d'une flèche pointant vers le haut à gauche (nw = nord-ouest).

text : curseur sous une forme qui symbolise du texte normal.

wait : curseur sous la forme d'un symbole signalant l'attente.

help : curseur sous forme d'un symbole qui signale de l'aide pour l'élément.

url(image), text; : curseur au choix, image doit être de type gif ou jpg.

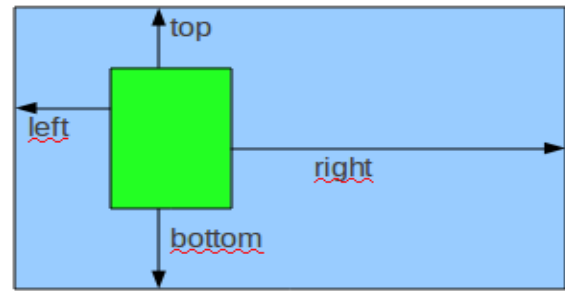
alias, all-scroll, auto, cell, context-menu, col-resize, copy, crosshair, default, e-resize, ew-resize, grab, grabbing, help, move, n-resize (n : north, e : east, s : south, w : west), ne-resize, nesw-resize, ns-resize, nw-resize, nwse-resize, no-drop, none, not-allowed, pointer, progress, row-resize, s-resize, se-resize, sw-resize, text,

url(img.gif), url(img.cur) on peut indiquer plusieurs, l'images doit être de petite taille. **Exemple** : `p{cursor : url("img1.gif"), url("second.jpg"), text; }`

### Propriétés de positionnement

**position** : permet de positionner au pixel près les éléments HTML (div, span, images etc ...). Valeurs :

absolute : pour le positionnement absolu, il est fait par



rapport au coin supérieur gauche de la fenêtre du navigateur à l'aide des propriétés auxiliaires

**left** : longueur (distance de la marge gauche de l'élément à la marge gauche de la fenêtre)

**right** : longueur ; **top** : longueur ou **bottom** : longueur. Exemple : `div{position: absolute; top: 80 px; left: 100 px;}`

**height** : définit la hauteur d'un élément. Valeurs : longueur, pourcentage du contenant, auto (valeur par défaut). Exemple : `p1{height: 50px}, p2{height: 50%}`.

**width** : définit la largeur d'un élément. Valeurs : longueur, pourcentage du contenant, auto (valeur par défaut). Exemple : `p1{width: 50px}, p2{width: 50%}`.

**min-height** : définit la hauteur minimale d'un élément. `img {height: 50%; min-height: 50px}`

**min-width** : définit la largeur minimale d'un élément. `img {width: 50%; width: 50px}`

**max-height et max-width** : Comme min-height et min-width, mais concerne la valeur maximale.

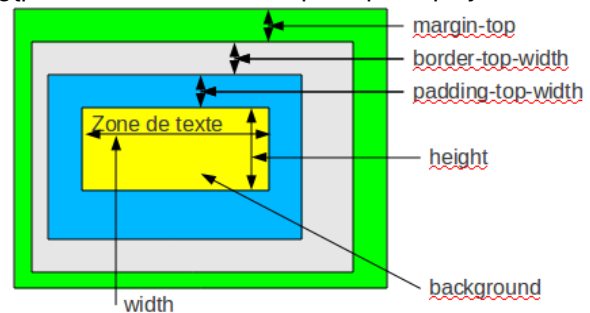
**Autres valeurs de la propriété position :**

static : valeur par défaut, les éléments apparaissent dans l'ordre.

fixed : positionne l'élément relativement à la fenêtre du navigateur.

relative : positionne l'élément relativement à sa position normale à l'aide des propriétés left et top.

`img{position:relative ; left:10px; top:100px;}`



Tous les éléments HTML s'inscrivent dans une boîte rectangulaire dont on peut définir les épaisseurs et les couleurs des bordures ainsi que les épaisseurs des paddings et des marges (gauche, droite, dessus, dessous), la couleur de fond... comme le montre l'image qui suit : des marges,

La visibilité du contenu d'un bloc est gouvernée par les propriétés suivantes :

**float** : s'applique au bloc qu'on veut faire flotter. (en faire couler du texte autour). Valeurs : none|left|right ; définit l'alignement d'un élément. Exemple : `img {float: left}`

**clear**: none | left | right | both ; empêche le flottement du côté indiqué de l'élément.

`img {float: left;} p{clear: both;}`  
 none : pas de règle spécifique  
**clip** : ne fait apparaître qu'une partie rectangulaire d'un élément (image) lorsque la propriété position a pour valeur absolue ou relative .  
`clip:rect(y1, x2, y2, x1)`  
 - x1 : valeur d'origine horizontale  
 - x2 : valeur horizontale maximale  
 - y1 : valeur d'origine verticale  
 - y2 : valeur verticale maximale  
`img {position:absolute; clip:rect(20px,100px,100px,20px);}`

**visibility** : visible | hidden | collapse affiche ou non l'élément. Collapse s'applique uniquement à table  
**display** : block | inline | list-item | none comment afficher certains éléments.  
 block : affiche l'élément dans une ligne  
 inline : affiche l'élément comme block  
 list-item : affiche l'élément comme un élément de liste  
**z-index** : nombre entier, dans le cas où des éléments sont partiellement ou totalement superposés, l'élément ayant le plus grand z-index est mis en avant plan et l'autre en arrière plan.

**overflow**: visible | hidden | scroll | auto  
 spécifie ce qui se passe quand le contenu déborde des dimensions allouées à l'élément. La valeur hidden cache ce qui déborde.

**overflow-x<sup>3</sup>** et **overflow-y<sup>3</sup>** ont même valeurs et effet que overflow mais uniquement horizontalement ou verticalement. Exemple :

`div {width:150px; height:200px; overflow-y:hidden;}`  
**vertical-align** : alignement vertical du contenu d'un élément. Valeurs : longueur (valeur négatives possibles), baseline (valeur par défaut), middle : centré verticalement, bottom, sub (indice), super (exposant), text-top alignement partie supérieure de l'élément, text-bottom alignement de la partie inférieure de l'élément. Pourcentage. Exemple : `img {vertical-align: bottom;}`  
`h1{position:absolute; width:6cm; height:1cm; top:50px; right:400px; background-color:red; visibility: visible;}`

### Propriétés des marges

**margin-top** : marge supérieure. Valeur : longueur, pourcentage ou auto. Exemple : `p1{ margin-top: 5px ;}` .  
**margin-right** : marge droite. Valeur : longueur, pourcentage ou auto.  
 Exemple : `p1{margin-right: 5px ;}` .  
**margin-bottom** : marge inférieure. Valeur : longueur, pourcentage ou auto.  
 Exemple : `img{ margin-bottom: 5px ;}` .  
**margin-left** : marge gauche. Valeur : longueur, pourcentage ou auto. Exemple : `p1{margin-left: 5px ;}` .  
**margin** : regroupe toutes les propriétés de marges ci-dessus. Exemple : `p{margin:2cm 4cm 3cm 4cm;}` dans l'ordre haut, droit, bas et gauche. `p{margin:2cm 4cm;}` indique haut, droit. `p{margin:2cm}` indique que toutes les marges sont à 2cm.

### Propriétés des Paddings

Le padding est la distance entre la zone de texte et la bordure d'un élément.

**padding-top** : padding haut. Valeur : longueur ou % `h3 {padding-top: 3px}`

**padding-right** : padding droite Valeur : longueur ou % `h3 {padding-right: 3px}`

**padding-bottom** : padding bas Valeur : longueur ou % `h3 {padding-bottom: 3px}`

**padding-left** : padding gauche Valeur : longueur ou % `h3 {padding-left: 3px}`

**padding** : peut prendre les différentes valeurs des propriétés des cotés des paddings. Peut avoir de 1 à 4 valeurs séparées par des espaces et dans l'ordre top, right, bottom et left.

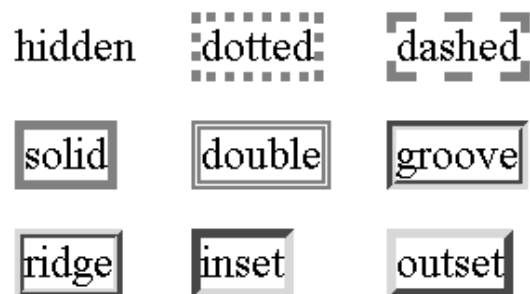
Exemple : `div{padding:10px 5px 15px ;}`

### Propriétés de bordures

Définit les caractéristiques des bordures des éléments (tableaux, cellules, boîtes...). On peut définir chaque côté séparément : bottom (dessous), top (dessus), right (Droite), left (gauche), Trois propriétés distinctes à chaque fois : width (largeur) style (style), color (couleur),

**border-bottom-color** : couleur du bord inférieur

**border-bottom-style** : style du bord inférieur valeurs possibles : none, solid, dotted, dashed, double, groove, ridge, inset ou outset.



**border-bottom-width** : épaisseur du bord inférieur

**border-bottom** : un résumé des propriétés précédentes

**border-bottom-left-radius<sup>3</sup>** : ; rayon de la bordure inférieure gauche. Valeurs : longueur , pourcentage.

**border-bottom-right-radius<sup>3</sup>** : longueur ; rayon de la bordure inférieure droite. Valeurs : longueur , pourcentage. `div {border-bottom-right-radius:50px;}`

Les propriétés précédentes sont valables avec top, left et right au lieu de bottom.

**border-top-left-radius<sup>3</sup>** : rayon de la bordure supérieure gauche.

**border-top-right-radius<sup>3</sup>** : rayon de la bordure supérieure droite.

**border-width** : regroupe les différentes propriétés d'épaisseur de bordure.

**border-color** : regroupe les différentes propriétés de couleur de bordure.

**border-style** : regroupe les différentes propriétés de style de bordure.

`{border-style: solid dashed ;}`

**Border** :regroupe toutes les propriétés de la bordure.

Exemple : `p{border:5px groove pink;}`

**border-image-outset<sup>3</sup>** : longueur, de dépassement de l'image de bordure de la bordure de l'élément.

**border-image-repeat**<sup>3</sup> : stretch|repeat|round  
stretch : image étirée, repeat : image répétée, round  
ajuste l'image pour couvrir entièrement chaque côté de la bordure.

**border-image-slice**<sup>3</sup> : *number*%|fill  
% relativement aux dimensions de l'image.

**border-image-source**<sup>3</sup> : url(border.png); l'image à utiliser comme bordure.

**border-image-width**<sup>3</sup> : épaisseur de l'image de bordure.

**border-image**<sup>3</sup> : regroupe toutes les propriétés de l'image de bordure.

**box-shadow**<sup>3</sup> : none|h-shadow v-shadow [blur] [spread] color |inset:

h-shadow : décalage horizontal de l'ombre

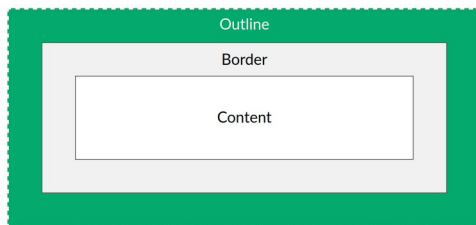
v-shadow : décalage vertical de l'ombre

blur : optionnel

spread : optionnel

color : couleur de l'ombre

### Propriétés de contour



**outline** est un contour autour d'un élément pour marquer l'extérieur. Il n'est pas possible de distinguer les cotés supérieur, gauche, droite et inférieur.

**outline-color** : couleur du contour.

**outline-style** : valeurs possibles : none, solid, dotted, dashed, double, groove, ridge, ridge, inset ou outset.

**outline-width** : largeur du contour.

**outline-offset**<sup>3</sup> : Espace entre la bordure de l'élément et son contour

**Outline** : regroupe les différentes propriétés précédentes.

**Exemple :**

```
p{outline-style:dotted; outline-color :red;}
```

### Pseudo-classes

En CSS Les pseudo-classes permettent d'accéder à certains éléments. Il constituent des classes prédéfinies contrairement aux classes. Les pseudo-classes permettent d'affiner le style appliqué à un certain nombre de balises en définissant une réaction à un événement ou bien à la position relative de la balise au sein des autres balises. Il existe plusieurs types de pseudo-classes :

#### pseudo-classes de liens

Pour chacun des états de lien on peut définir une mise en forme particulière, à l'aide des pseudo classes

**a:link** : lien

**a:visited** : lien visité

**a:active** : lien sous-clique

**a:hover** : curseur survole le lien

Exemple : `a:hover{color:red; text-decoration:none;}`

### Pseudo-classes pour toutes les balises

**:hover** : Définit un style particulier pour tout élément survolé.

**:active** : Définit un style particulier pour un lien activé.

**:focus** : Définit un style particulier pour un lien ou un élément de formulaire ayant le focus.

```
input:focus:hover {font-size: 12pt; color: red;}
```

```
table:hover{background:red;}
```

### Pseudo-éléments :

**:first-letter** : définit un style particulier pour la première lettre d'un texte.

Exemple : `P:first-letter { font-size: 200%; font-weight: bold; }`

**:first-line** : finit un style particulier pour la première ligne d'un texte.

```
p:first-line { text-transform: uppercase }
```

**:before** Insère un contenu avant celui d'un élément.

```
p:before{content:"Paragraphe : ";
```

**:after** Insère un contenu après celui d'un élément.

Exemple : `p:after { content:"- Remember this";`

```
background-color:yellow;
```

```
color:red;
```

```
font-weight:bold; }
```

Exemple : `h1:before {content: counter(chapno, upper-roman) ". "}`

### Priorité

La concurrence entre plusieurs éléments de style peut provenir des différentes possibilités de localisation de feuilles de style :

- dans un fichier externe avec l'extension .css.

- dans la balise head du document.

- dans le body du document.

La règle de priorité est d'appliquer la feuille de style la plus proche de l'élément par le navigateur .

Il y a cependant moyen de contourner cette règle de priorité par la déclaration **!important**; par exemple :

```
body { background-color : #0000FF !important;}
```

### Héritage en CSS

Certaines propriétés CSS sont transmises d'éléments ancêtres à des éléments descendants. On appelle cela l'héritage (inheritance).

```
ol {color:#FF0000;}
```

li hérite de ol

```
body {font-family: Arial, sans-serif; font-size: 12px;}
```

Ces propriétés s'appliquent à tous les descendants de body.

Toutes les propriétés ne sont pas héritées par exemple : border, margin etc

### Compteurs

content : s'utilise avec :before et :after pour insérer un contenu.

Counter-increment : incrémente un ou plusieurs compteurs.

counter-reset : crée ou remet à zéro un ou plusieurs compteurs.

quotes : Sets the type of quotation marks for embedded quotations

**Exemple :**

```
body{counter-reset:section;}
```



```
h1{counter-reset:subsection;}
h1:before{
counter-increment:section;
content:"Section " counter(section) ". ";}
h2:before{
counter-increment:subsection;
content:counter(section) "." counter(subsection) ". ";}
```

### Arborescence d'un fichier (X)HTML

Un document HTML est un arbre, avec des ancêtres, des descendants, des parents et des enfants

**descendant** : Un élément qui est l'enfant, le petit enfant ou plus éloigné encore d'un élément dans l'arbre du document.

**ancêtre** : Un élément qui est le parent, le grand parent ou plus éloigné encore d'un élément dans l'arbre du document.[c]

**enfant** : Le descendant direct d'un élément. Aucun autre élément ne peut s'intercaler entre les deux dans l'arbre du document.

**parent** : L'ancêtre direct d'un élément. Aucun autre élément ne peut s'intercaler entre les deux dans l'arbre du document.

### Sélecteurs

#### Sélection simultanée

**S1, S2, ..., Sn {propriété1:valeur1; ... }** déclaration commune à plusieurs sélecteurs (qui sont, séparés par des virgules) . Exemple : h1, h2, h3, h4 {color:red ;}.

#### Sélecteur de descendance : S1 S2

**S1 S2 {propriété1:valeur1; ... }** (sélecteurs séparés par un espace) la déclaration s'applique à S2 uniquement quand celui-ci est un descendants de S1.

Exemple : h1 code {...} la déclaration s'applique uniquement aux balises <code> ... </code> qui sont dans les titres </h1>.

ol li {...} les éléments li dans les listes

#### Sélecteur d'enfant : S1 > S2

**S1 > S2 {propriété1:valeur1; ... }** la déclaration s'applique à S2 uniquement quand celui-ci est un enfant de S1.

Exemple : td > ul{...}

Déclaration s'applique uniquement à <ul> ... </ul> contenue dans une cellule <td> ... </td> de table.

#### Sélecteur de succession : S1 + S2

**S1 + S2 {propriété1:valeur1; ... }**

la déclaration s'applique à S2 uniquement quand celui-ci est placé juste après S1.

Exemple : h1+p {color:red}

<h1> ... </h1>

<p> ... </p>

<p> ... </p>

Seul le premier paragraphe sera en rouge.

**S1 ~ S2** : Sélectionne tout élément S2 placé après un élément S1 dans la page

**S1 >> S2** : Sélectionne tous les éléments S2 à l'intérieur des éléments S1.

#### Sélecteur Universel : \*

applique les règles CSS à tous les éléments de la page, même les propriétés de bloc

\* {border-width : 0; } : supprimer toutes les bordures par défaut.

.ma\_classe est équivalent à \*.ma\_classe

\* {margin:0; padding:0;} remet à zéro les propriétés margin et padding de tous les éléments :

#### Sélecteur d'attribut

**balise[attribut]{propriété1:valeur1; ... }** la déclaration modifie le style de <Balise> uniquement si elle contient l'attribut attribut. (La balise cible doit contenir l'attribut cité). Exemple : table[border] {border-style: double } <table border="...">

**[attr]** cible les éléments qui possèdent un attribut attr.

**[attr=valeur]** cible les éléments qui possèdent un attribut attr et dont la valeur de cet attribut est exactement valeur.

**[attr~=valeur]** cible les éléments qui possèdent un attribut attr et dont la valeur de cet attribut une liste de mots séparés par des blancs dont un vaut exactement valeur.

**[attr]=valeur]** cible les éléments qui possèdent un attribut attr et dont la valeur de cet attribut est exactement valeur ou dont la valeur commence par valeur suivi immédiatement d'un tiret.

**[attr^=valeur]** cible les éléments qui possèdent un attribut attr et dont la première valeur commence par valeur.

**[attr\$=valeur]** cible les éléments qui possèdent un attribut attr et dont la dernière valeur termine par valeur.

**[attr\*=valeur]** cible les éléments qui possèdent un attribut attr et dont la valeur contient au moins une occurrence d'une chaîne de caractères qui contient valeur.

### Pseudo-classes de page

Le sélecteur **@page** permet de modifier les définitions de mise en page d'une page HTML (taille, marge, etc.) à l'impression, telles que les marges (margin-left, margin-top, margin-right, margin-bottom), la taille

**@page:left** : permet de définir les propriétés des pages de gauche.

**@page:left** { size: landscape; margin-left: 2cm; }

**@page:right** : permet de définir les propriétés des pages de droite.

**@page:right** { size:landscape; margin-left: 2.5cm; }

**@page:first** : permet de définir les propriétés de la première page d'un document.

**@page:first** { size: portrait; }