

HTML & CSS

Cours 3 :

La mise en forme avec CSS

Nous allons voir la théorie sur le CSS : qu'est-ce que c'est et où est-ce qu'on écrit du code CSS.

1) Mettre en place le CSS

1.1) La petite histoire du CSS

CSS (Cascading Style Sheets), c'est cet autre langage qui vient compléter le HTML.

1.1.1) Petit rappel : à quoi sert CSS ?

C'est le CSS qui vous permet de choisir la couleur de votre texte, de sélectionner la police utilisée sur votre site, de définir la taille du texte, les bordures, le fond, etc...

Et aussi, c'est lui qui permet de faire la mise en page de votre site. Vous pourrez dire : je veux que mon menu soit à gauche et occupe telle largeur, que l'en-tête de mon site soit calé en haut et qu'il soit toujours visible, etc...

Grâce au HTML, nous rédigeons le contenu de notre site mais il est brut de décoffrage.

Le CSS vient compléter ce code pour mettre en forme tout cela et donner au contenu l'apparence que l'on souhaite.

1.1.2) CSS : des débuts difficiles

Il faut savoir qu'aux débuts du Web, CSS n'existait pas. En fait, il n'y avait initialement que le langage HTML.

Le HTML est né en 1991 et CSS en 1996. Alors, vous vous dites sûrement : comment faisait-on la mise en forme de 1991 à 1996 ? Eh bien, uniquement en HTML ! Il y avait en effet des balises HTML dédiées à la mise en forme. ``, par exemple, permettait de définir la couleur du texte.

Cependant, les pages HTML commençaient à devenir assez complexes. Il y avait de plus en plus de balises et c'était un joyeux mélange entre le fond et la forme, qui rendait la mise à jour des pages web de plus en plus complexe. C'est pour cela que l'on a créé le langage CSS.

Cependant, le CSS n'a pas été adopté immédiatement par les webmasters, loin de là. Il fallait se défaire de certaines mauvaises habitudes et cela a pris du temps. Encore aujourd'hui, on peut trouver des sites web avec des balises HTML de mise en forme, anciennes et obsolètes, comme ``.

1.1.3) CSS : la prise en charge des navigateurs

Tout comme le HTML, le CSS a évolué. Il y a quatre versions importantes de CSS :

- CSS 1
- CSS 2.0
- CSS 2.1
- CSS 3

En fait, la version CSS 3 n'est pas encore totalement finalisée (ce n'est pas encore une version officielle). Cependant, elle est bien avancée et aujourd'hui déjà bien prise en charge par de nombreux navigateurs, ce qui fait qu'on peut déjà s'en servir.

Il serait dommage de passer à côté car CSS 3 apporte de nombreuses fonctionnalités à CSS (leur nombre double par rapport à CSS 2.1 !). Nous nous baserons donc dans ce cours sur CSS 3, qui reprend et complète la plupart des fonctionnalités de CSS 2.1.

Ce sont les navigateurs web qui font le travail le plus complexe : ils doivent lire le code CSS et comprendre comment afficher la page.

Au début des années 2000, Internet Explorer était le navigateur le plus répandu mais sa gestion du CSS est longtemps restée assez médiocre (pour ne pas dire carrément mauvaise). C'était la grande époque de la version 6 (IE6), hélas encore utilisée aujourd'hui par une petite partie des internautes (heureusement, cette proportion tend à diminuer).

Que faut-il retenir de tout cela ?

Que les navigateurs ne connaissent pas toutes les propriétés CSS qui existent. Plus le navigateur est vieux, moins il connaît de fonctionnalités CSS.

Nous allons voir dans ce cours un certain nombre de fonctionnalités de CSS qui ne marchent pas forcément sur les navigateurs les plus vieux.

Au pire, si le navigateur ne connaît pas une propriété CSS, il l'ignore et ne met pas en forme, mais cela ne fait pas planter votre page : celle-ci sera donc toujours lisible.

Je vous recommande fortement de mettre dans vos favoris les sites www.caniuse.com et normansblog.de qui proposent des tables de compatibilité des fonctionnalités de HTML et CSS sur différents navigateurs (et sur leurs différentes versions). Regardez en particulier les tables de compatibilité pour CSS de www.caniuse.com.

1.2) Où écrit-on le CSS ?

Vous avez le choix car on peut écrire du code en langage CSS à trois endroits différents :

- dans un fichier .css (méthode la plus recommandée)
- dans l'en-tête <head> du fichier HTML
- directement dans les balises du fichier HTML via un attribut style (méthode la moins recommandée).

Nous allons voir ces trois méthodes mais sachez d'ores et déjà que la première... est la meilleure.

1.2.1) Externe : dans un fichier « .css » (recommandé)

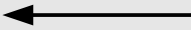
On écrit le plus souvent le code CSS dans un fichier spécial ayant l'extension « .css ». C'est la méthode la plus pratique et la plus souple. Cela nous évite de tout mélanger dans un même fichier.

Exemple :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <link rel="stylesheet" href="style.css" />
    <title>Premiers tests du CSS</title>
  </head>

  <body>
    <h1>Mon super site</h1>

    <p>Bonjour et bienvenue sur mon site !</p>
    <p>Pour le moment, mon site est un peu <em>vide</em>. Patientez encore un
    peu !</p>
  </body>
</html>
```



Vous noterez le contenu de la ligne 5,

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

c'est elle qui indique que ce fichier HTML est associé à un fichier appelé « style.css ».

Voici le code CSS contenu dans style.css :

```
p {  
    color: blue;  
}
```

Cela a pour conséquence de mettre tous le texte des paragraphes en bleu.

Il est inutile d'ouvrir directement le fichier « style.css » dans le navigateur. Il faut ouvrir le fichier page.html (il fera automatiquement appel au fichier style.css).

1.2.2) Interne : dans l'en-tête <head> du fichier HTML

Il existe une autre méthode pour utiliser du CSS dans ses fichiers HTML : cela consiste à insérer le code CSS directement dans une balise <style> à l'intérieur de l'en-tête <head>.

Voici comment on peut obtenir exactement le même résultat avec un seul fichier .html qui contient le code CSS (lignes 5 à 10)

```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8" />  
    <style>  
      p {  
        color: blue;  
      }  
    </style>  
    <title>Premiers tests du CSS</title>  
  </head>  
  
  <body>  
    <h1>Mon super site</h1>  
  
    <p>Bonjour et bienvenue sur mon site !</p>  
    <p>Pour le moment, mon site est un peu <em>vide</em>. Patientez encore un  
peu !</p>  
  </body>  
</html>
```

1.2.3) Inline : dans les balises (non recommandé)

Dernière méthode, à manipuler avec précaution vous pouvez ajouter un attribut « style » à n'importe quelle balise. Vous insérerez votre code CSS directement dans cet attribut :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <title>Premiers tests du CSS</title>
  </head>

  <body>
    <h1>Mon super site</h1>

    <p style="color: blue;">Bonjour et bienvenue sur mon site !</p>
    <p>Pour le moment, mon site est un peu <em>vide</em>. Patientez encore un
peu !</p>
  </body>
</html>
```

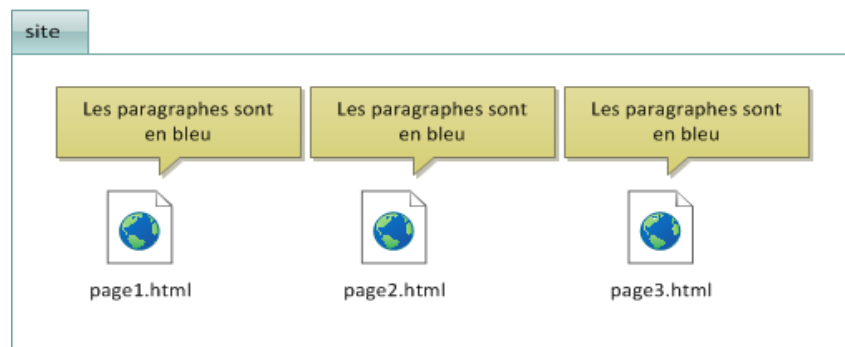
Cette fois, seul le texte du premier paragraphe (ligne 11), dont la balise contient le code CSS, sera coloré en bleu.

1.2.4) Quelle méthode choisir ?

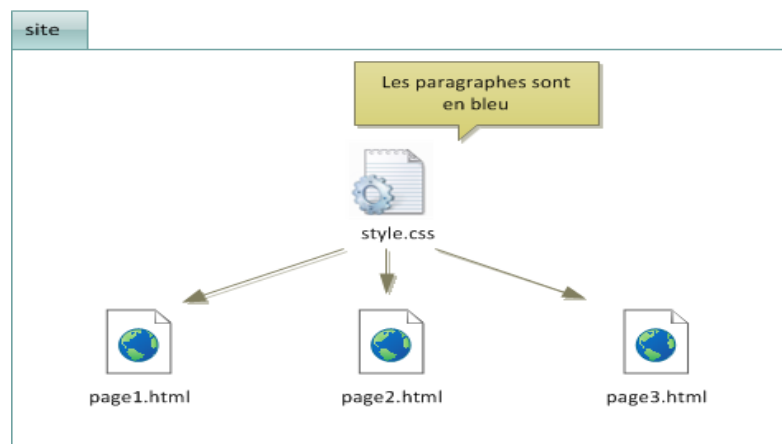
Je vous recommande fortement de prendre l'habitude de travailler avec la première méthode parce que c'est celle utilisée par la majorité des webmasters... Pourquoi ?

Pour le moment, vous faites vos tests sur un seul fichier HTML. Cependant, votre site sera plus tard constitué de plusieurs pages HTML, on est d'accord ?

Imaginez : si vous placez le code CSS directement dans le fichier HTML, il faudra copier ce code dans tous les fichiers HTML de votre site ! Et si demain vous changez d'avis, par exemple si vous voulez que vos paragraphes soient écrits en rouge et non en bleu, il faudra modifier chaque fichier HTML un à un, comme le montre la figure suivante :



Le code CSS est répété dans chaque fichier HTML



Le code CSS est donné une fois pour toutes dans un fichier CSS

Si vous travaillez avec un fichier CSS externe, vous n'aurez besoin d'écrire cette instruction qu'une seule fois pour tout votre site, comme le montre la figure suivante :

1.3) Appliquer un style : sélectionner une balise

Maintenant que nous savons où placer le code CSS, intéressons-nous de plus près à ce code. Reprenons le bout de code CSS vu précédemment :

```
p {  
    color: blue;  
}
```

Dans un code CSS comme celui-ci, on trouve trois éléments différents :

Des noms de balises : on écrit les noms des balises dont on veut modifier l'apparence.

Par exemple, si je veux modifier l'apparence de tous les paragraphes <p>, je dois écrire « p ».

Des propriétés CSS : les « effets de style » de la page sont rangés dans des propriétés.

Il y a par exemple la propriété « color » qui permet d'indiquer la couleur du texte, « font-size » qui permet d'indiquer la taille du texte, etc...

Les valeurs : pour chaque propriété CSS, on doit indiquer une valeur.

Par exemple, pour la propriété « color », il faut indiquer le nom de la couleur. Pour « font-size », il faut indiquer quelle taille on veut, etc...

Schématiquement, une feuille de style CSS ressemble donc à cela :

```
balise1 {  
    propriete1: valeur1;  
    propriete2: valeur2;  
}  
  
balise2 {  
    propriete1: valeur1;  
    propriete2: valeur2;  
    propriete3: valeur3;  
}  
  
balise3 {  
    propriete1: valeur1;  
}
```

Vous repérez dans cet extrait de code les balises, propriétés et valeurs dont je viens de vous parler.

Comme vous le voyez, on écrit le nom de la balise (par exemple h1) et on ouvre des accolades pour, à l'intérieur, mettre les propriétés et valeurs que l'on souhaite.

On peut mettre autant de propriétés que l'on veut à l'intérieur des accolades. Chaque propriété est suivie du symbole « deux-points » (:) puis de la valeur correspondante. Enfin, chaque ligne se termine par un point-virgule (;).

Le code vu précédemment :

```
p {  
  color: blue;  
}
```

... signifie donc en français : «Je veux que tous mes paragraphes soient écrits en bleu».

1.3.1) Appliquer un style à plusieurs balises

Prenons le code CSS suivant :

```
h1 {  
  color: blue;  
}  
  
em {  
  color: blue;  
}
```

Il signifie que nos titres <h1> et nos textes importants doivent s'afficher en bleu. Par contre, c'est un peu répétitif. Heureusement, il existe un moyen en CSS d'aller plus vite

Si les deux balises doivent avoir la même présentation. Il suffit de combiner la déclaration en séparant les noms des balises par une virgule

comme ceci :

```
h1, em {  
  color: blue;  
}
```

Cela signifie : « Je veux que le texte de mes <h1> et soit écrit en bleu ». Vous pouvez indiquer autant de balises à la suite que vous le désirez.

1.3.2) Des commentaires dans du CSS

Comme en HTML, il est possible de mettre des commentaires. Les commentaires ne seront pas affichés, ils servent simplement à indiquer des informations pour vous, par exemple pour vous y retrouver dans un long fichier CSS.

D'ailleurs, vous allez vous en rendre compte, en général le fichier HTML est assez court et la feuille CSS assez longue (si elle contient tous les éléments de style de votre site, c'est un peu normal). Notez qu'il est possible de créer plusieurs fichiers CSS pour votre site si vous ressentez le besoin de séparer un peu votre code CSS (en fonction des différentes sections de votre site, par exemple).

Donc, pour faire un commentaire :

Écrivez /*, suivi de votre commentaire, puis */ pour terminer votre commentaire.

Vos commentaires peuvent être écrits sur une ou plusieurs lignes.

Exemple :

```
/*  
style.css  
-----  
  
Par Eleve BTS IRIS  
*/  
  
p {  
    color: blue; /* Les paragraphes seront en bleu */  
}
```

1.4) Appliquer un style : class et id

Ce que je vous ai montré jusqu'ici a quand même un défaut : cela implique par exemple que TOUS les paragraphes possèdent la même présentation (ici, ils seront donc tous écrits en bleu).

Comment faire pour que certains paragraphes seulement soient écrits d'une manière différente ?

On pourrait placer le code CSS dans un attribut style sur la balise que l'on vise (c'est la technique que je vous ai présentée un peu plus tôt) mais, comme je vous l'ai dit, ce n'est pas recommandé (il vaut mieux utiliser un fichier CSS externe).

Pour résoudre le problème, on peut utiliser ces attributs spéciaux qui fonctionnent sur toutes les balises :

l'attribut class

l'attribut id

Que les choses soient claires dès le début : les attributs class et id sont quasiment identiques. Il y a seulement une petite différence que je vous dévoilerai plus bas.

Pour le moment, et pour faire simple, on ne va s'intéresser qu'à l'attribut class.

Comme je viens de vous le dire, c'est un attribut que l'on peut mettre sur n'importe quelle balise, aussi bien titre que paragraphe, image, etc...

```
<h1 class=""> </h1>
<p class=""> </p>
<img class="" />
```

Vous devrez écrire un nom qui sert à identifier la balise. Ce que vous voulez, du moment que le nom commence par une lettre.

Par exemple, je vais associer la classe introduction à mon premier paragraphe :

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset="utf-8" />
    <link rel="stylesheet" href="style.css" />
    <title>Premiers tests du CSS</title>
  </head>

  <body>
    <h1>Mon super site</h1>

    <p class="introduction">Bonjour et bienvenue sur mon site !</p>
```

```
<p>Pour le moment, mon site est un peu <em>vide</em>. Patientez encore un  
peu !</p>  
</body>  
</html>
```

Le paragraphe est identifié par un nom : « introduction ».
Vous allez pouvoir réutiliser ce nom dans le fichier CSS

pour dire : «Je veux que seules les balises qui ont comme nom 'introduction' soient affichées en bleu».

Pour faire cela en CSS, indiquez le nom de votre classe en commençant par un point, comme ci-dessous :

```
.introduction {  
    color: blue;  
}
```

Et l'attribut id alors ?

Lui, il fonctionne exactement de la même manière que « class », à un détail près : il ne peut être utilisé qu'une fois dans le code.

Quel intérêt ? Cela sera utile pour le JavaScript pour reconnaître certaines balises. D'ailleurs, nous avons déjà vu l'attribut « id » dans le chapitre sur les liens (pour réaliser des ancrés). En pratique, nous ne mettrons des id que sur des éléments qui sont uniques dans la page, comme par exemple le logo :

```

```

Si vous utilisez des « id », lorsque vous définirez leurs propriétés dans le fichier CSS, il faudra faire précéder le nom de « l'id » par un dièse (#) :

```
#logo {  
    /* Indiquez les propriétés CSS ici */  
}
```

Récapitulatif :

Avec l'attribut « class » plusieurs balises peuvent avoir le même nom.

Un nom « d'id » doit être unique dans la page HTML.

1.4.1) Les balises universelles

Il arrivera parfois que vous ayez besoin d'appliquer une class (ou un id) à certains mots qui, à l'origine, ne sont pas entourés par des balises.

En effet, le problème de class, c'est qu'il s'agit d'un attribut. Vous ne pouvez donc en mettre que sur une balise. Si, par exemple, je veux modifier uniquement « bienvenue » dans le paragraphe suivant :

```
<p>Bonjour et bienvenue sur mon site !</p>
```

Cela serait facile à faire s'il y avait une balise autour de « bienvenue » mais, malheureusement il n'y en a pas. Par chance, on a inventé... la balise-qui-ne-sert-à-rien.

En fait, on a inventé **deux balises dites universelles, qui n'ont aucune signification particulière** (elles n'indiquent pas que le mot est important, par exemple). Il y a une différence minime (mais significative !) entre ces deux balises :

** : c'est une balise de type inline, c'est-à-dire une balise que l'on place au sein d'un paragraphe de texte, pour sélectionner certains mots uniquement. Cette balise s'utilise au milieu d'un paragraphe**

Les balises et sont de la même famille.

<div> </div> :c'est une balise de type block, qui entoure un bloc de texte.

Les balises <p>, <h1>, etc. sont de la même famille. Ces balises ont quelque chose en commun : elles créent un nouveau « bloc » dans la page et provoquent donc obligatoirement un retour à la ligne. <div> est une balise fréquemment utilisée dans la construction d'un design, comme nous le verrons plus tard.

Exemple d'utilisation de la balise span :

```
<p>Bonjour et <span class="salutations">bienvenue</span> sur mon site !</p>
```

```
salutations {  
    color: blue;  
}
```

1.5) Appliquer un style : les sélecteurs

En CSS, le plus difficile est de savoir cibler le texte dont on veut changer la forme. Pour cibler (on dit « sélectionner ») les éléments de la page à modifier, on utilise ce qu'on appelle des **sélecteurs**.

Ces sélecteurs, que nous avons vus précédemment, sont de loin les plus couramment utilisés. **Il faut les connaître par cœur.**

```
p { ... }
```

Signifie « Je veux toucher tous les paragraphes ».

```
h1, em { ... }
```

Signifie « Tous les titres et tous les textes importants ». Nous avons sélectionné deux balises d'un coup.

Et enfin, nous avons vu comment sélectionner des balises précises à qui nous avons donné un nom grâce aux attributs « class » et « id » :

```
.class { ... }
```

```
#id { ... }
```

Il existe des dizaines d'autres façons de cibler des balises en CSS ! Nous n'allons pas toutes les voir car il y en a beaucoup et certaines sont complexes, mais voici déjà de quoi vous permettre d'être plus efficaces en CSS !

1.6) En résumé

- CSS est un autre langage qui vient compléter le HTML. Son rôle est de mettre en forme votre page web.
- Il faut être vigilant sur la compatibilité des navigateurs avec certaines fonctionnalités récentes de CSS3. Quand un navigateur ne connaît pas une instruction de mise en forme, il l'ignore simplement.
- On peut écrire le code CSS à plusieurs endroits différents, le plus conseillé étant de créer un fichier séparé portant l'extension .css (exemple : style.css).
- En CSS, on sélectionne quelles portions de la page HTML on veut modifier et on change leur présentation avec des propriétés CSS :

2) Formatage du texte

Cela signifie simplement que l'on va modifier l'apparence du texte (on dit qu'on le « met en forme »).

Ce chapitre va être l'occasion de découvrir de nombreuses propriétés CSS : nous allons voir comment modifier la taille du texte, changer la police, aligner le texte...

2.1) « font-size » : La taille

Pour modifier la taille du texte, on utilise la propriété CSS « font-size ».

Plusieurs techniques vous sont proposées :

- Indiquer une **taille absolue** :

en pixels, en centimètres ou millimètres.

- Indiquer une **taille relative** :

en pourcentage, « em » ou « ex ».

2.1.1) Une taille absolue

Pour indiquer une taille absolue, on utilise généralement les pixels.

Exemple :

```
p {  
  font-size: 14px; /* Paragraphes de 14 pixels */  
}  
h1 {  
  font-size: 40px; /* Titres de 40 pixels */  
}
```

2.1.2) Taille relative

Il y a plusieurs moyens d'indiquer une valeur relative. Vous pouvez par exemple écrire la taille avec des mots en anglais comme ceux-ci :

- xx-small : minuscule
- x-small : très petit
- small : petit
- medium : moyen
- large : grand

- x-large : très grand
- xx-large : gigantesque

Exemple :

```
p {
  font-size: small;
}
h1 {
  font-size: large;
}
```

Cette technique a un défaut : il n'y a que sept tailles disponibles (car il n'y a que sept noms). Heureusement, il existe d'autres moyens.

La technique recommandée consiste à indiquer la taille en « em ».

- Si vous écrivez 1em, le texte a une taille normale.
- Si vous voulez grossir le texte, vous pouvez inscrire une valeur supérieure à 1, comme 1.3em.
- Si vous voulez réduire le texte, inscrivez une valeur inférieure à 1, comme 0.8em.

Attention : pour les nombres décimaux, il faut mettre un point et non une virgule.

Vous devez donc écrire « 1.4em » et non pas « 1,4em »

Exemple :

```
p {
  font-size: 0.8em;
}
h1 {
  font-size: 1.3em;
}
```

D'autres unités sont disponibles. Vous pouvez essayer le « ex » (qui fonctionne sur le même principe que le em mais qui est plus petit de base) et le pourcentage (80%, 130% ...).

2.2) « font-family » : Modifier la police utilisée

Il se pose un problème : pour qu'une police s'affiche correctement, il faut que tous les internautes l'aient. Si un internaute n'a pas la même police que vous, son navigateur prendra une police par défaut (une police standard) qui n'aura peut-être rien à voir avec ce à quoi vous vous attendiez.

La propriété CSS qui permet d'indiquer la police à utiliser est « font-family ».

Vous devez écrire le nom de la police comme ceci :

```
balise {  
    font-family: nomPolice;  
}
```

Seulement, pour éviter les problèmes si l'internaute n'a pas la même police que vous, on précise en général plusieurs noms de police, séparés par des virgules :

```
balise {  
    font-family: police1, police2, police3, police4;  
}
```

Le navigateur essaiera d'abord d'utiliser la police1. S'il ne l'a pas, il essaiera la police2. S'il ne l'a pas, il passera à la police3, et ainsi de suite. En général, on indique en tout dernier « serif », ce qui correspond à une police par défaut (qui ne s'applique que si aucune autre police n'a été trouvée).

Il existe aussi une autre police par défaut appelée « sans-serif ». La différence entre les deux est la présence de petites pattes de liaison en bas des lettres, que la police « sans-serif » n'a pas.

Voici une liste de polices qui fonctionnent bien sur la plupart des navigateurs :

- Arial
- Arial Black
- Comic Sans MS
- Courier New
- Georgia
- Impact
- Times New Roman
- Trebuchet MS
- Verdana

Exemple :

```
p {  
    font-family : Impact, "Arial Black", Arial, Verdana, sans-serif;  
}
```

Cela signifie : « Mets la police Impact ou, si elle n'y est pas, Arial Black, ou sinon Arial, ou sinon Verdana, ou si rien n'a marché, mets une police standard (sans-serif) ».

En général, il est bien d'indiquer un choix de trois ou quatre polices (+ serif ou sans-serif) afin de s'assurer qu'au moins l'une d'entre elles aura été trouvée sur l'ordinateur du visiteur.

Remarque : Si le nom de la police comporte des espaces, il est conseillé de l'entourer de guillemets, comme pour Arial Black.

2.3) Italique, gras, souligné...

Il existe en CSS une série de propriétés classiques de mise en forme du texte. Nous allons découvrir ici comment afficher le texte en gras, italique, souligné... et au passage nous verrons qu'il est même possible d'aller jusqu'à le faire clignoter !

2.3.1) « font-style » : Mettre en italique

En CSS, pour mettre en italique, on utilise « font-style »

qui peut prendre trois valeurs :

- **italic** : le texte sera mis en italique
- **oblique** : le texte sera passé en oblique (les lettres sont penchées, le résultat est légèrement différent de l'italique proprement dit).
- **normal** : le texte sera normal (par défaut). Cela vous permet d'annuler une mise en italique. Par exemple, si vous voulez que les textes entre `` ne soient plus en italique, vous devrez écrire

```
em {  
  font-style: normal;  
}
```

Exemple : je me sers de font-style pour mettre en italique tous mes titres `<h2>`

```
h2 {  
  font-style: italic;  
}
```

2.3.2) « font-weight » : Mettre en gras

La propriété CSS pour mettre en gras est « font-weight »

et prend les valeurs suivantes :

- **bold** : le texte sera en gras
- **normal** : le texte sera écrit normalement (par défaut)

Exemple : Ecrire les titres en gras

```
h1 {  
  font-weight: bold;  
}
```

2.3.3) « text-decoration » : Soulignement et autres décorations

La propriété CSS associée porte bien son nom : « text-decoration ».

Elle permet, entre autres, de souligner le texte, mais pas seulement. Voici les différentes valeurs qu'elle peut prendre :

- **underline** : souligné
- **line-through** : barré
- **overline** : ligne au-dessus
- **blink** : clignotant. Ne fonctionne pas sur tous les navigateurs (Internet Explorer et Google Chrome, notamment).
- **none** : normal (par défaut)

Ce CSS va vous permettre de tester les effets de « text-decoration » :

```
h1 {  
    text-decoration: blink;  
}  
.souligne  
{  
    text-decoration: underline;  
}  
.barre  
{  
    text-decoration: line-through;  
}  
.ligne_dessus  
{  
    text-decoration: overline;  
}
```

2.4) « text-align » : L'alignement

Le langage CSS nous permet de faire tous les alignements connus : à gauche, centré, à droite et justifié.

On utilise la propriété « text-align » et on indique l'alignement désiré :

- **left** : le texte sera aligné à gauche (c'est le réglage par défaut).
- **center** : le texte sera centré
- **right** : le texte sera aligné à droite
- **justify** : le texte sera « justifié ». Justifier le texte permet de faire en sorte qu'il prenne toute la largeur possible sans laisser d'espace blanc à la fin des lignes. Les textes des journaux, par exemple, sont toujours justifiés.

```
h1 {  
  text-align: center;  
}  
  
p {  
  text-align: justify;  
}  
  
.signature {  
  text-align: right;  
}
```

Remarque : Vous ne pouvez pas modifier l'alignement du texte d'une balise inline (comme , <a>, , ...). L'alignement ne fonctionne que sur des balises de type block (<p>, <div>, <h1>, <h2>, ...) et c'est un peu logique, quand on y pense : on ne peut pas modifier l'alignement de quelques mots au milieu d'un paragraphe !

C'est donc en général le paragraphe entier qu'il vous faudra aligner.

3) La couleur et le fond

Continuons notre tour d'horizon des propriétés CSS existantes. Nous allons nous intéresser ici aux propriétés liées de près ou de loin à la couleur. Nous verrons entre autres :

- comment changer la couleur du texte
- comment mettre une couleur ou une image d'arrière-plan
- comment ajouter des ombres
- comment jouer avec les niveaux de transparence

3.1) *Couleur du texte « color »*

Vous connaissez déjà la propriété qui permet de modifier la couleur du texte : il s'agit de « color ». Nous allons nous intéresser aux différentes façons d'indiquer la couleur, car il y en a plusieurs.

3.1.1) Indiquer le nom de la couleur

La méthode la plus simple et la plus pratique pour choisir une couleur consiste à taper son nom (en anglais). Le seul défaut de cette méthode est qu'il n'existe que seize couleurs dites « standard » :

White	Silver	Gray	Black
Red	Maroon	Lime	Green
Yellow	Olive	Blue	Navy
Fuchsia	Purple	Aqua	Teal

Les seize noms de couleurs utilisables en CSS

Exemple : Pour passer tous les titres en marron

```
h1 {  
    color: maroon;  
}
```

3.1.2) La notation hexadécimale

Seize couleurs, c'est quand même un peu limite quand on sait que la plupart des écrans peuvent en afficher seize millions.

Heureusement, il existe en CSS plusieurs façons de choisir une couleur parmi toutes celles qui existent. La première que je vais vous montrer est la notation hexadécimale. Elle est couramment utilisée sur le Web mais il existe aussi une autre méthode que nous verrons plus loin.

Un nom de couleur en hexadécimal, cela ressemble à : #FF5A28.

On doit toujours commencer par écrire un dièse (#), suivi de six lettres ou chiffres allant de 0 à 9 et de A à F. Ces lettres ou chiffres fonctionnent deux par deux.

Les deux premiers indiquent une quantité de rouge, les deux suivants une quantité de vert et les deux derniers une quantité de bleu.

En mélangeant ces quantités (qui sont les composantes Rouge-Vert-Bleu de la couleur) on peut obtenir la couleur qu'on veut.

Ainsi, #000000 correspond à la couleur noire et #FFFFFF à la couleur blanche.

Certains logiciels de dessin, comme Photoshop, Gimp et Paint.NET, vous indiquent les couleurs en hexadécimal. Il vous est alors facile de copier-coller le code hexadécimal d'une couleur dans votre fichier CSS.

Pour appliquer aux paragraphes la couleur blanche en hexadécimal :

```
p {  
  color: #FFFFFF;  
}
```

Remarque : Il existe une notation raccourcie, on peut écrire une couleur avec seulement trois caractères. Par exemple : #FA3 équivaut à écrire #FFAA33.

3.1.3) La méthode RGB

Que signifie RGB ? En anglais, Rouge-Vert-Bleu s'écrit Red-Green-Blue, ce qui s'abrège en « RGB ». Comme avec la notation hexadécimale, pour choisir une couleur, on doit définir une quantité de rouge, de vert et de bleu, (mais cette fois-ci en décimal).

```
P {  
  color: rgb(240,96,204);  
}
```

Ces quantités sont toujours comprises entre 0 et 255.

3.2) Couleur de fond « background-color »

Pour indiquer une couleur de fond, on utilise la propriété CSS background-color. Elle s'utilise de la même manière que la propriété color, c'est-à-dire que vous pouvez taper le nom d'une couleur, l'écrire en notation hexadécimale ou encore utiliser la méthode RGB.

Pour indiquer la couleur de fond de la page web, il faut travailler sur la balise <body>. Eh oui, <body> correspond à l'ensemble de la page web, c'est donc en modifiant sa couleur de fond que l'on changera la couleur d'arrière-plan de la page :

```
/* On travaille sur la balise body, donc sur TOUTE la page */  
body {  
    background-color: black; /* Le fond de la page sera noir */  
    color: white; /* Le texte de la page sera blanc */  
}
```

3.3) Images de fond « background-image »

Tout comme pour la couleur de fond, n'oubliez pas que l'image de fond ne s'applique pas forcément à la page entière. On peut aussi mettre une image de fond derrière les titres, paragraphes, etc...

3.3.1) Appliquer une image de fond

La propriété permettant d'indiquer une image de fond est « background-image ». Comme valeur, on doit renseigner url("nom_de_l_image.png").

Exemple :

```
body  
{  
    background-image: url("neige.png");  
}
```

Bien entendu, le fond n'est pas forcément en PNG, il peut aussi être en JPEG ou en GIF (voir cours sur les images).

L'adresse indiquant où se trouve l'image de fond peut être écrite en absolu (http://...) ou en relatif (fond.png) (voir cours sur les liens).

Remarque :

Lorsque vous écrivez une adresse en relatif dans le fichier CSS, l'adresse de l'image doit être indiquée par rapport au fichier .css et non pas par rapport au fichier .html.

Pour simplifier les choses, je vous conseille de placer l'image de fond dans le même dossier que le fichier .css (ou dans un sous-dossier).

4) Le CSS et l'héritage

En CSS, si vous appliquez un style à une balise, toutes les balises qui se trouvent à l'intérieur prendront le même style.

C'est en fait simple à comprendre et intuitif. La balise `<body>`, vous le savez, contient entre autres les balises de paragraphe `<p>` et de titre `<h1>`.

Si j'applique une couleur de fond noire et une couleur de texte blanche à la balise `<body>`, tous mes titres et paragraphes auront eux aussi un arrière-plan de couleur noire et un texte de couleur blanche...

C'est ce phénomène qu'on appelle l'héritage :

on dit que les balises qui se trouvent à l'intérieur d'une autre balise « héritent » de ses propriétés.

C'est d'ailleurs de là que vient le nom « CSS », qui signifie « Cascading Style Sheets », c'est-à-dire « Feuilles de style en cascade ».

Les propriétés CSS sont héritées en cascade : si vous donnez un style à un élément, tous les sous-éléments auront le même style.

Cela ne fonctionne pas uniquement pour la couleur.

Toutes les propriétés CSS seront héritées.

Vous pouvez par exemple demander une mise en gras dans la balise `<body>` et tous vos titres et paragraphes seront en gras.

Exemple d'héritage avec la balise <mark> :

On a tendance à croire qu'on ne peut modifier que la couleur de fond de la page. C'est faux : vous pouvez changer le fond de n'importe quel élément : vos titres, vos paragraphes, certains mots... Dans ce cas, ils apparaîtront surlignés (comme si on avait mis un coup de marqueur dessus).

Code HTML :

```
<h1>Qui a éteint la lumière ?</h1>

<p>Brr, il fait tout noir sur ce site, c'est un peu <mark>inquiétant</mark>
comme ambiance non vous trouvez pas ?</p>
```

Code CSS :

```
body {
    background-color: black;
    color: white;
}

mark {
    /* La couleur de fond prend le pas sur celle de toute la page */
    background-color: red;
}
```

Sur le texte de la balise <mark>, c'est la couleur de fond rouge qui s'applique.

C'est la propriété CSS de l'élément le plus précis qui a la priorité.

Le même principe vaut pour toutes les balises HTML et toutes les propriétés CSS. Si vous dites :

- mes paragraphes ont une taille de 1.2 em
- mes textes importants () ont une taille de 1.4 em

On pourrait penser qu'il y a un conflit. Le texte important fait partie d'un paragraphe, quelle taille lui donner ? 1.2 em ou 1.4 em ? Le CSS décide que c'est la déclaration la plus précise qui l'emporte : comme correspond à un élément plus précis que les paragraphes, le texte sera écrit en 1.4 em.

5) Les bordures

Nous allons nous intéresser aux bordures et aux effets d'ombrage que l'on peut appliquer, aussi bien sur le texte que sur les blocs qui constituent notre page.

5.1) *Bordures standard « border »*

Le CSS vous offre un large choix de bordures pour décorer votre page. De nombreuses propriétés CSS vous permettent de modifier l'apparence de vos bordures : border-width, border-color, border-style...

Pour aller à l'essentiel, je vous propose ici d'utiliser directement la super-propriété « border » qui regroupe l'ensemble de ces propriétés.

Pour border on peut utiliser jusqu'à trois valeurs pour modifier l'apparence de la bordure :

- **La largeur** : indiquez la largeur de votre bordure. Mettez une valeur en pixels (comme 2px).
- **La couleur** : c'est la couleur de votre bordure. Utilisez, comme on l'a appris, soit un nom de couleur (black, red,...), soit une valeur hexadécimale (#FF0000), soit une valeur RGB (rgb(198, 212, 37)).
- **Le type de bordure** : Voici les différentes valeurs disponibles :
 - none : pas de bordure (par défaut)
 - solid : un trait simple
 - dotted : pointillés
 - dashed : tirets
 - double : bordure double
 - groove : en relief
 - ridge : autre effet relief
 - inset : effet 3D global enfoncé
 - outset : effet 3D global surélevé

Ainsi, pour avoir une bordure bleue, en tirets, épaisse de 3 pixels autour de mes titres, je vais écrire :

Code CSS :

```
h1 {  
    border: 3px blue dashed;  
}
```

5.1.1) En haut, à droite, à gauche, en bas...

Il est possible de mettre des bordures différentes en fonction du côté (haut, bas, gauche ou droite), vous pouvez le faire sans problème. Dans ce cas, vous devrez utiliser ces quatre propriétés :

- `border-top` : bordure du haut
- `border-bottom` : bordure du bas
- `border-left` : bordure de gauche
- `border-right` : bordure de droite

Il existe aussi des équivalents pour paramétrer chaque détail de la bordure si vous le désirez : `border-top-width` pour modifier l'épaisseur de la bordure du haut, « `border-top-color` » pour la couleur du haut, etc...

Ce sont aussi des super-propriétés, elles fonctionnent comme `border` mais ne s'appliquent donc qu'à un seul côté.

Pour ajouter une bordure uniquement à gauche et à droite des paragraphes, on écrira donc :

Code CSS :

```
p {  
  border-left: 2px solid black;  
  border-right: 2px solid black;  
}
```

Remarque : On peut modifier les bordures de n'importe quel type d'élément sur la page. Nous l'avons fait ici sur les paragraphes mais on peut aussi modifier la bordure des images, des textes importants comme ``, etc...

5.2) Bordures arrondies « *border-radius* »

C'est une propriété nouvelle du CSS3

La propriété « `border-radius` » va permettre d'arrondir facilement les angles de n'importe quel élément. Il suffit d'indiquer la taille (« l'importance ») de l'arrondi en pixels :

Code CSS :

```
p {  
  border-radius: 10px;  
}
```

On peut aussi préciser la forme de l'arrondi pour chaque coin. Dans ce cas, indiquez quatre valeurs

Code CSS :

```
p {  
  border-radius: 10px 5px 10px 5px;  
}
```

Les valeurs correspondent aux angles suivants dans cet ordre :

1. en haut à gauche

2. en haut à droite
3. en bas à droite
4. en bas à gauche

Enfin, il est possible d'affiner l'arrondi de nos angles en créant des courbes elliptiques. Dans ce cas, il faut indiquer deux valeurs séparées par une barre oblique (slash, caractère /).

Code CSS :

```
p {  
    border-radius: 20px / 10px;  
}
```

Remarque :

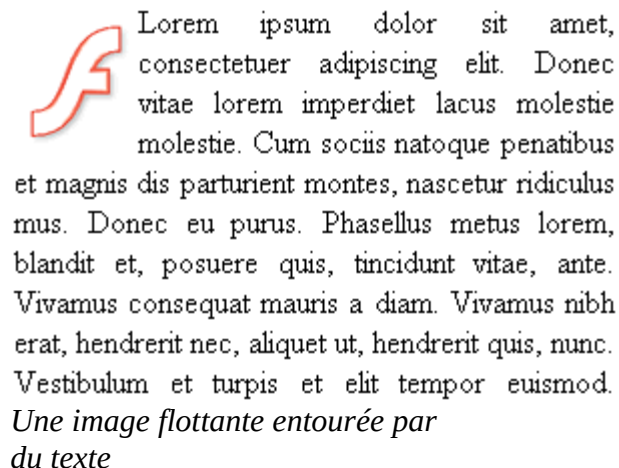
Les bordures arrondies fonctionnent avec tous les navigateurs récents, y compris Internet Explorer à partir de la version 9 (IE9).

Pour les anciennes versions de Mozilla Firefox, Chrome et Safari, il était nécessaire d'utiliser ce qu'on appelle des « préfixes vendeurs », c'est-à-dire qu'il fallait écrire dans le code CSS différentes versions de la propriété (-moz-border-radius pour Firefox, -webkit-border-radius pour Safari, etc.). Ce n'est heureusement plus nécessaire aujourd'hui, sauf si vous voulez gérer les anciennes versions de ces navigateurs.

6) « float » : Les flottants

Le CSS nous permet de faire flotter un élément autour du texte. On dit aussi qu'on fait un « habillage »

Pour que vous voyiez bien de quoi on parle, la figure suivante vous montre ce que nous allons apprendre à faire.



Cela se fait en utilisant la propriété « float » (flottant en anglais). Cette propriété peut prendre deux valeurs très simples

- **left** : l'élément flottera à gauche
- **right** : l'élément flottera à droite

L'utilisation des flottants est très simple :

1. Vous appliquez un float à une balise
2. Puis vous continuez à écrire du texte à la suite normalement

On peut aussi bien utiliser la propriété « float » sur des balises « block » que sur des balises « inline ».

Il est courant de faire flotter une image pour qu'elle soit habillée par du texte, comme dans l'exemple précédent.

6.1) Faire flotter une image

Voici le code HTML :

```
<p>  
   Ceci est  
  un texte normal de paragraphe, écrit à la suite de l'image et qui l'habillera  
  car l'image est flottante.  
</p>
```

Vous devez placer l'élément flottant en premier dans le code HTML.

Si vous placez l'image après le paragraphe, l'effet ne fonctionnera pas.

Voici le code CSS :

```
.imageflottante {  
  float: left;  
}
```

6.2) « clear » : Stopper un flottant

Si l'on veut qu'au bout d'un moment le texte continue en dessous du flottant comme dans figure suivante :



Il existe en fait une propriété CSS qui permet de dire : « Stop, ce texte doit être en-dessous du flottant et non plus à côté ». C'est la propriété « clear », qui peut prendre ces trois valeurs :

- **left** : le texte se poursuit en-dessous après un « float :left »
- **right** : le texte se poursuit en-dessous après un « float : right »
- **both** : le texte se poursuit en-dessous, que ce soit après un « float: left » ou après un « float : right »

Pour simplifier, on va utiliser tout le temps le « clear : both », qui marche après un flottant à gauche et après un flottant à droite (cela fonctionne donc à tous les coups).

Pour illustrer son fonctionnement, on va prendre ce code HTML :

```
<p></p>  
<p>Ce texte est écrit à côté de l'image flottante.</p>  
<p class="dessous">Ce texte est écrit sous l'image flottante.</p>
```

Et ce code CSS :

```
.imageflottante {  
    float: left;  
}  
.dessous {  
    clear: both;  
}
```

On applique un « clear : both » au paragraphe que l'on veut voir continuer sous l'image flottante.

Notes

<http://www.siteduzero.com/tutoriel-3-13517-mettre-en-place-le-css.html>

Table des matières

1)Mettre en place le CSS.....	1
1.1)La petite histoire du CSS.....	1
1.1.1)Petit rappel : à quoi sert CSS ?.....	1
1.1.2)CSS : des débuts difficiles.....	2
1.1.3)CSS : la prise en charge des navigateurs.....	2
1.2)Où écrit-on le CSS ?.....	4
1.2.1)Externe : dans un fichier « .css » (recommandé).....	4
1.2.2)Interne : dans l'en-tête <head> du fichier HTML.....	5
1.2.3)Inline : dans les balises (non recommandé).....	6
1.2.4)Quelle méthode choisir ?.....	7
1.3)Appliquer un style : sélectionner une balise.....	8
1.3.1)Appliquer un style à plusieurs balises.....	9
1.3.2)Des commentaires dans du CSS.....	10
1.4)Appliquer un style : class et id.....	11
1.4.1)Les balises universelles.....	13
1.5)Appliquer un style : les sélecteurs.....	14
1.6)En résumé.....	14
2)Formatage du texte.....	15
2.1)« font-size » : La taille.....	15
2.1.1)Une taille absolue.....	15
2.1.2)Taille relative.....	15
2.2)« font-family » : Modifier la police utilisée.....	16
2.3)Italique, gras, souligné.....	18
2.3.1)« font-style » : Mettre en italique.....	18
2.3.2)« font-weight » : Mettre en gras.....	18
2.3.3)« text-decoration » : Soulignement et autres décorations.....	19
2.4)« text-align » : L'alignement.....	20
3)La couleur et le fond.....	21
3.1)Couleur du texte « color ».....	21
3.1.1)Indiquer le nom de la couleur.....	21
3.1.2)La notation hexadécimale.....	21
3.1.3)La méthode RGB.....	22
3.2)Couleur de fond « background-color ».....	23
3.3)Images de fond « background-image ».....	23
3.3.1)Appliquer une image de fond.....	23
4)Le CSS et l'héritage.....	24
5)Les bordures.....	26
5.1)Bordures standard « border ».....	26
5.1.1)En haut, à droite, à gauche, en bas.....	26

5.2) Bordures arrondies « border-radius ».....	27
6) « float » : Les flottants.....	29
6.1) Faire flotter une image.....	30
6.2) « clear » : Stopper un flottant.....	30